Towards a Vision Algorithm Compiler for Recognition of Partially Occluded 3-D Objects

Mark D. Wheeler Katsushi Ikeuchi

November 20, 1992
CMU-CS-92-185

School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213

This research was sponsored by the Avionics Laboratory, Wright Research and Development Center, Aeronautical Systems Division (AFSC), U. S. Air Force, Wright-Patterson AFB, OH 45433-6543 under Contract F33615-90-C-1465, Arpa Order No. 7597. The first author was supported by a National Science Foundation Graduate Fellowship. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the Defense Advanced Research Projects Agency, the U.S. Government, or the National Science Foundation.

Keywords: computer vision, 3-D object recognition, Markov random fields, pose estimation, localization, robust estimation

Abstract

Our goal is to develop a model-based vision system that is capable of recognizing 3-D objects in range images in spite of partial occlusion of the objects. We present new methods for object recognition and localization and describe the implementation and performance of these methods in our Vision Algorithm Compiler (VAC) model-based vision system. The VAC is given a sensor model and a set of geometric object models and generates a recognition/localization program for the specified objects in images from the specified sensor. Our recognition algorithm is based on the hypothesize-and-verify paradigm. We use the *sensor-modeling* approach to build accurate models of our prior-knowledge constraints that account for constraints due to sensor characteristics, feature-extraction algorithm behavior, model geometry, and the effects of partial occlusion. We phrase the hypothesis-generation process as a search for the most likely set of hypotheses based on our prior knowledge–in contrast to typical constrained combinatorial searches. The Markov random field (MRF) formalism and Highest Confidence First estimation [Cho88] provide us with an efficient and effective technique for performing this search. Our algorithm utilizes a robust localization and verification procedure that is accurate in spite of partial occlusion of the object. Our results demonstrate the ability of our localization algorithm to accurately localize models in complicated scenes with partial occlusion. To demonstrate the utility of our robust localization algorithm, our results are compared with those of the commonly used least-squares methods. The ability of our recognition algorithm to recognize objects while limiting the number of verifications is demonstrated on several test images.

1. Introduction

The task of object recognition is to describe the objects present in an image. The most successful object-recognition methods belong to the model-based vision paradigm. In this paradigm, the system is given models of a set of known objects and an image, and its task is to recognize and locate any of the known objects present in the image. The model-based vision problem is typically phrased as a matching problem. The models are represented as a set of primitive features which correspond to features that can be extracted from an input image. The problem is to find a matching or correspondence between the model features and image features which determines the location of the object in the scene.

The problem can be approached using intensity images (2-D images) or depth/range images (3-D images). The type of image used determines the primitive features that can be used to do the matching. For 2-D images, features based on photometric properties must be used. With 3-D images, geometric features are available. The availability of depth information does not solve the problem of recognition but simply adds another dimension to the constraints we may apply. The availability of high quality range image sensors has spurred research in object recognition from 3-D data [FH86, BHH87, GLP87, IK88, Fan90, IH91, KK91, FJ91, SM92]. As the techniques for generating range images improve and sensor hardware becomes widely available, the potential applications of 3-D object-recognition algorithms will greatly increase.

At this time, several model-based vision systems have been developed for 2-D images as well as 3-D images. The majority of these systems are based on the *hypothesize-and-verify* paradigm—a correspondence of image and model primitives (matches) is hypothesized, and the image is tested against a projection of the hypothesized object into the image to determine if the hypothesis is accepted or rejected. The ability of systems to generate accurate hypotheses depends greatly on the prior knowledge available. Most systems only model the geometry of the object and do not account for the effects of the sensor characteristics, the feature-extraction algorithm, or partial occlusion. Without an accurate model of sensor and segmentation characteristics, the hypothesis-generation procedure must compensate for the inaccuracies by loosening the constraints and, thus, increasing the number of incorrect hypotheses that are generated. A common difficulty is the ability to extract primitive features from images reliably. Many object-recognition systems require unrealistic capabilities of typically unreliable feature-extraction procedures. Such requirements limit the applicability of these systems. In order to recognize partially occluded objects, many systems rely on local point or edge features which are especially difficult to reliably extract from cluttered scenes. We believe that for recognition algorithms to be effective, they must use accurate models of known constraints and must minimize their reliance on feature-extraction algorithms.

We present a novel approach to object recognition that addresses the above problems. Our approach to hypothesis generation is "optimal" in terms of the number of hypotheses that require verification with respect to our prior knowledge of the objects, sensor, and feature-extraction routines. The *sensor-modeling* approach is used to build prior models that account for constraints due to sensor characteristics, feature-extraction-algorithm behavior,

model geometry, and the effects of partial occlusion. Here, hypothesis generation is phrased as a search for the most likely set of hypotheses based on our prior knowledge instead of the typical constrained combinatorial search. The formalism of Markov random fields (MRF) and Highest Confidence First estimation [Cho88] provide us with an efficient and effective technique for performing this search. The use of accurate models of our prior-knowledge constraints enables us to minimize the generation of incorrect hypotheses.

Our verification algorithm utilizes a new method for 3-D object localization given a rough initial estimate of the object's position. The algorithm is driven by the range data and does not rely on matches between high-level image features and model features. This method refines the pose estimate through energy minimization in a manner similar to deformable templates, active contours, and snakes [TWK88, KWT87]. The novelty of our algorithm is the use of a robust estimator for the energy function being minimized. The robust estimator is insensitive to outliers occurring due to partial occlusion of the object as well as other effects of complicated scenes. Most previous methods effectively employ least-squares techniques to solve for model parameters. These methods have poor performance when applied to images where there is much partial occlusion or when other objects are near the object of interest.

In this report, we describe the implementation and performance of our recognition algorithm in our Vision Algorithm Compiler (VAC) model-based vision system for recognizing polyhedral models in range images. Our results demonstrate the ability of our localization algorithm to localize models in complicated scenes with partial occlusion. To demonstrate the utility of our robust localization algorithm, our results are compared with those of the commonly used least squares methods. The ability of our algorithm to recognize objects while limiting the number of verifications is demonstrated on several test images.

We begin by discussing related work. Our approach and motivations are described in Section 3. Section 4 describes the MRF formulation for hypothesis generation. Our localization and verification algorithm is described in Section 5. In Section 6, the time complexity of our algorithms is analyzed. In Section 7, we look at some recognition and localization results of our algorithm. Section 8 summarizes the contributions of this work, and Section 9 details areas for improvement of our system and proposed future work.

2. Related Work

Here, we briefly describe some of the influential efforts in the field of model-based vision that are related to our work. First, we would like to discuss some of the work on 3-D object-recognition systems from range images. For the most part, these systems can be classified as hypothesize-and-verify systems. They vary in the constraints and primitive features used, the types of objects modeled, and the style of search over the space of hypotheses.

- Bolles, Horaud, and Hannah [BHH87] describe the 3DPO system for recognizing industrial parts in 3-D images. Their approach is to grow matches by searching for feature matches that will add the most information to the current interpretation, thereby reducing the degrees of freedom in the interpretation. Initially, a "focus feature" is chosen (one that is likely to provide the most constraints on the interpretation), and a cluster of features is grown around this focus feature and matched to model features of the current interpretation. The features used by 3DPO are range-image edges augmented by information about the surfaces adjacent to the edge.

- Faugeras and Hebert [FH86] developed a system to recognize and locate 3-D rigid models in 3-D images. Their features include planar patches, polygonal edge chains and characteristic points. Their method utilizes the rigidity constraint to determine whether a set of matches is compatible. Thus, they are able to perform recognition and localization simultaneously. The rigidity constraint is also used to guide the search process to find consistent matchings.

- Grimson and Lozano-Perez [GLP87] developed an interpretation-tree based, object-recognition and localization system for overlapping parts. They investigated the use of decoupled and coupled constraints and a preprocessing step of coarse Hough clustering. Coupled constraints, used to enforce global consistency of the current interpretation, significantly increased the computation while not significantly improving the pruning of the search tree compared to decoupled constraints. Hough clustering was used as a coarse filter to produce an initial set of model-data pairings. It effectively reduced the size of the interpretation tree to be searched but had the effect of adding more recognition failures.

- Ikeuchi, Kanade and Hong [IK88, IH91] describe a Vision Algorithm Compiler designed to localize a known model in a bin of parts. They utilize a sensor model to determine the set of aspects of the object. An aspect is defined to be an equivalence class of possible images of the object with respect to the sensor model and essentially specifies the set of visible image features of the object from a set of viewing directions. Their system produces an executable interpretation tree which first classifies the aspect of the topmost object in the bin. The classified aspect guides the localization computation of the object. Since the visible image features of a model are known to vary only slightly within a given aspect, a configuration space approach is used to determine the most likely set of matching edges which are then used to compute the location of the object. Their method assumes that the objects are not partially occluded.

- Kim and Kak [KK91] describe a system for 3-D object recognition in 3-D images using surface and edge features. For a given model hypothesis, they construct a bipartite graph of possible image feature to model feature matches. Their requirement for a correct match is that it is complete (all image features match at least one model feature) and injective (each image feature map to a different model feature). They utilize bipartite graph matching to find complete matchings of the graph and in turn reject hypothesized models when a complete matching is not found. Discrete relaxation is used to prune matches that are inconsistent based on relational constraints. This method requires that image features can be grouped into sets belonging to single objects.

- Hutchinson, Cromwell and Kak [HCK89] describe a method of applying uncertainty reasoning to model-based object recognition. They use Dempster-Shafer theory to generate the belief values of hypothesized scene-model matches. They generate training samples to estimate the parameters of a Gaussian distribution describing the probability that a given set of points came from a cylindrical, planar or spherical surface. They also show that the Dempster combination rule can be executed in polynomial time for their system. Unfortunately, their method is exponential in the number of sensed features.

- Fan [Fan90] developed a recognition system based on 3-D surface descriptions. The segmentation algorithm is based on identifying extremal curves on the object surfaces and uses a multiscale approach to detect these cleanly. Segmented patches are grouped to form a relational graph structure which is the basic object representation. His system utilizes automatic model acquisition to acquire characteristic views of the model-base objects. The recognition process then attempts to match the segmented graph structures with the stored model-view structures. The requirement of grouping patches into sets belonging to a single object is compensated by a split operation and a merge operation. The split operation divides a graph into smaller subgraphs when the complete graph cannot be matched. The merge operation uses a successful recognition of an object to identify other surfaces belonging to the object that were not correctly grouped with its other recognized surfaces.

- Another variation on hypothesis generation is to use descriptive features which allow direct indexing into a table of the model features to generate a match hypothesis. Stein and Medioni [SM92] introduce the use of *splashes* in a model-based vision system for 3-D object recognition. Splashes are local features computed on a geodesic coordinate system and are particularly well suited for recognition rigid models composed of free formed surfaces. Their locality makes them useful for recognition in spite of occlusion. In addition to splashes, their system uses a viewpoint-invariant representation of connected 3-D edges called 3-D curves. Both the splash and 3-D curve features are represented by 3-D super-segments which are composed of linked 3-D edges represented at multiple scales. The super-segments are encoded to allow indexing into a table to provide fast model retrieval and matching. A single super-segment match is, in most cases, sufficient fo. pose estimation. Verification consists of finding clusters of hypotheses which obey rigidity constraints.

A couple of related techniques, geometric hashing and the alignment method, use a brute force approach to the hypothesize-and-verify paradigm.

- Lamdan and Wolfson [LW88] present a general recognition technique called geometric hashing for recognition of 3-D models from 2-D and 3-D images. Geometric hashing is a voting scheme where a minimal set of features are chosen to form a basis function, and the coordinates of all image features are calculated in this basis. An off-line process creates a hash table of the possible coordinates of all model features with respect to all model basis sets. The coordinate of each image feature indexes into the hash table which contains a list of the basis sets that could produce that coordinate, and a vote for each of these basis sets is recorded. If a basis receives a high number of votes, the basis defines the model and location of the hypothesis which is then verified using the standard project-and-match technique.

- The alignment technique developed by Huttenlocher and Ullman [Hut88] is also based on the concept of utilizing interest points in the models and images to form the basis of a coordinate frame. Here, a set of image points are selected as a basis and, by matching them to model points, are used to transform the model into the scene. The projected model features are compared with image features to verify the existence of the hypothesized object location specified by the basis. Thus, the localization is implicit in the recognition.

Constraint-satisfaction techniques have been utilized to perform object recognition and feature description by several researchers.

- Bolle, Califano and Kjeldsen [BCK90] describe an object-recognition system based on levels of abstraction from low-level data to high-level features and objects that are linked by parameter transforms. Bottom-up connections provide the input at each level of abstraction, and intra-level connections provide the constraints between hypotheses. Constraint satisfaction is used to arrive at consistent interpretations of the representation at each level.

- Cooper [Coo89] developed a system for recognizing Tinker-Toy world objects. His method modeled the objects and image primitives in a Markov random field (MRF) and used a non-optimal search method (Highest Confidence First [Cho88]) to find good interpretations of the scene. The system was able to recognize objects from 2-D images in spite of occlusion. However, the model-bases tested consisted of only a few models. In his method, there are problems with the time and space requirements for even small model bases. The MRF must contain a field for each model for each possible occurrence in the image.

- Parvin and Medioni [PM] describe a constraint-satisfaction network for matching surfaces of 3-D objects. Simulated annealing of a Boltzman network is used to determine an optimal matching satisfying local (matching of individual surfaces), adjacency

(maintain geometric consistency between adjacent surfaces) and global constraints (ensure matching is satisfied under a rigid-body transform). This work demonstrates that it is feasible to utilize constraint-satisfaction networks to match surfaces with known 3-D models. The intent was not to perform recognition though the basic techniques are applicable.

Several researchers have used a dynamic approach to localizing a known or hypothesized model in the scene. The central idea is the minimization of some metric or energy function to solve for the unknown model parameters.

- Yuille, Cohen, and Hallinan [YCH89] introduce the use of templates to locate and estimate the parameters of face features from 2-D images. They use a gradient-descent search over an energy function defined over low-level image features to solve for the model parameters.

- Lowe [Low85, Low89] developed the SCERPO object-recognition system for intensity images which uses perceptual grouping of image features and matching of the grouped features to grouped model features to generate an initial pose estimate. His localization method matches image and model features (edges) and minimizes the least-squared error over the model parameters. He uses a technique based on Newton-Raphson root-finding and Levenberg-Marquardt minimization to iteratively compute the model parameters. He developed an object modeling system designed for efficient computation of visibility of model features over the model parameters. This makes it possible for his system to predict the appearance of model features quickly enough to perform localization in real time motion sequences. He has applied his localization technique to object tracking in image sequences and has developed a system for tracking the motion of parameterized objects.

- Sato and Ikeuchi [SIK92] describe a model-based vision system for recognizing the aspect of a known specular object from 2-D images. They describe their models using aspects over the appearance of specular features. Their approach uses a Dempster-Shafer theory formulation of hypothesis generation. They take the most likely hypotheses and perform deformable template matching to compute the verification metric. The hypothesis with the lowest energy is accepted as the recognized aspect of the object.

- Delingette, Hebert and Ikeuchi [DHI92] developed a method of fitting a deformable tesselated sphere to 3-D range data. Their method was based on a dynamics formulation over the model parameter space with respect to energy functions defined by depth and intensity edges, the range data, and model smoothness. The objects that can be extracted must have the same topology as a sphere, but other topological structures could possibly be applied as the base structure. The smoothness energy limits the ability of the model to match rigid structures with sharp edges and introduces the problem of selecting smoothness parameters for the model fit.

- Pentland and Sclaroff [PS91] describe a physically-based, dynamics formulation for solving for deformation parameters of objects modeled by finite element surfaces. Their solution utilizes modal analysis to reduce the dimensions of the systems to be solved.
- Terzopoulos and Metaxas [TM91] present a physically-based approach to solve for the superquadric parameters of an object in a range or intensity image.
- Besl and McKay [BM92] describe a method called the "iterative closest point" algorithm which efficiently computes the pose of a 3-D shape given a set of 3-D points that belong to the shape. Their method assumes that the number of outliers is near zero which requires that all of the image data belong to the given model.

3. Approach

Our recognition algorithm is based on the *hypothesize-and-verify* paradigm which pervades model-based vision. The hypothesis-generation phase produces a set of hypothesized matches between model primitives and image primitives. The verification phase then measures whether the image data adequately supports the hypotheses or not. If there is sufficient evidence of correspondence between the hypothesized model and the image data, the model is recognized as being in the image.

The goal of the hypothesis-generation phase is to throw out as many of the wrong hypotheses and keep as many of the correct hypotheses as possible. For each hypothesis produced, a comparatively expensive verification step is required. The hypothesis-generation phase must use prior knowledge to filter out incorrect hypotheses to reduce the amount of verification steps needed to recognize the objects in the scene. The execution time of a hypothesize-and-verify recognition program is critically dependent on the number of hypotheses that have to be verified. There are two ways that the hypothesis-generation procedure can reduce the number of hypotheses to be verified: accurate *selection* of hypotheses and optimal *ordering* of hypotheses for verification.

If the hypothesis selection is accurate, all of the correct hypotheses are selected while the number of incorrect hypotheses selected is minimized. The selection of hypotheses for verification is often based on the quality of match between viewpoint-invariant model features and image features. In most systems, the selection process uses constraints that are based solely on the geometric models of the objects and do not account for sensor or feature-extraction-algorithm characteristics. Without an accurate model of sensor and segmentation characteristics, the hypothesis-selection procedure must compensate for the inaccuracies by loosening the constraints and, thus, increasing the number of incorrect hypotheses that are generated. Hypothesis generation should be "optimal" with respect to our prior knowledge of the objects, sensor, and feature-extraction routines. Our solution, the *sensor-modeling* approach, is to build accurate prior models of the constraints due to sensor and segmentation characteristics in addition to model geometry. We formulate the task of "optimal" hypothesis selection as a search for the most likely set of matches based on our prior knowledge. Essentially, we are making a decision to only attempt to verify the hypotheses that justify, based on our prior knowledge, the expense of the verification step. This is accomplished by integrating observed image features and our prior-knowledge constraints in the formalism of Markov random fields (MRF). With the MRF formulation, the search for the most likely hypotheses is phrased as a *maximum a posteriori* (MAP) estimate of the MRF. To compute the MAP estimate exactly would be prohibitively expensive. We utilize a sub-optimal estimation procedure called Highest Confidence First (HCF) [Cho88] which is efficient in practice and finds good (useful) estimates of the most likely hypotheses. In complex scenes, the selection of hypotheses is complicated by totally or partially occluded object features. The sensor-modeling approach enables us to model these effects by simulating partial occlusion of the objects when generating the prior models.

A successful verification of a hypothesis eliminates other competing hypotheses from

consideration. Therefore, a good ordering of the hypotheses for verification can reduce the number of hypotheses requiring verification. Traditionally, hypotheses are ordered based on the saliency (discrimination ability) or size of the image features. These heuristics have proven useful for hypothesis ordering; however, to minimize verifications, we want to first verify the most likely hypotheses—not necessarily those with the most salient or largest image features. Our solution is to order the hypotheses for verification by their likelihood based on our prior knowledge. Thus, our hypothesis-generation method is "optimal", with respect to our prior knowledge, in terms of ordering hypotheses for verification.

Accurate localization is crucial for reliable object recognition. Almost all object-recognition systems perform a verification step to determine if a particular object is present in the image. If the location estimate is inaccurate, the verification will fail, and the object may not be recognized. A good localization method should be robust with respect to errors due to partial occlusion and features arising from other objects in the scene.

There are two main drawbacks to the the majority of localization techniques described in Section 2: occlusion sensitivity and reliance on matches between high-level image features and model features. The first problem, occlusion sensitivity, results from the sensitivity of the location estimation algorithms to outliers from the model. For the problem of localization in range images, an outlier is any range-data point that doesn't belong to the model being localized or any model point that is not visible from the current viewing direction. The outlier can be caused by occlusion by another object or an object that is sufficiently close to the object of interest to attract the model to the wrong object. The sensitivity of the estimation procedure to outliers is directly dependent on the assumptions about the error distributions expected in the images. If the outliers are assumed to be highly unlikely, the presence of a few outliers will ruin the estimate.

The second issue, reliance on primitive-feature matches, is a problem because high-level features can be very noisy and difficult to reliably compute in complicated scenes. Noise in the features or missing features can lead to inaccurate pose estimates. The existence of multiple objects and partial occlusion exacerbates the problem. With occlusion, the high-level matches may leave the pose estimate underconstrained. Slight variations of the object dimensions from that of the model will cause the location estimates based on high-level matches to be inaccurate. More often, the high-level matches provide a good initial estimate of the pose which can be improved by tweaking the estimate a bit to match the low-level scene data. What we need is a method which will allow us to drop our model into parameter space using our initial estimate and let the 3-D image data act to pull the model to a more accurate estimate—the essential idea of active contours and snakes [TWK88, KWT87].

To solve the above problems, we developed a new variant of template matching ([YCH89]) for model-parameter refinement and localization of 3-D models in range images. The template consists of points sampled from the surface of the object model. The template points are attracted to points in the range image and allow us to solve for the pose by finding a minimum-energy state of the template in the model-parameter space. Our formulation assumes an error distribution which assumes that outliers are likely. This makes the approach

robust with respect to outliers (such as those caused by occlusion). Our method is driven directly by the image data and not by features computed by an unreliable or unpredictable feature-extraction procedure.

Part of our philosophy in designing this system was to minimize our dependence on feature-extraction algorithms. Our system only relies on 3-D surfaces as the primitive feature for matching. Surfaces are easier to discriminate than point and edge features, which makes them useful for efficient matching. We describe the surfaces by unary and binary (relational) features. The reluctance of other researchers to utilize surfaces alone is due to the fact that self-occlusion of the surface makes the unary features viewpoint-dependent. However, our sensor-modeling approach builds prior models that account for the effects of self-occlusion and partial occlusion; thus, we never assume that the surfaces are completely visible when generating the prior models. Our approach makes these "viewpoint-dependent" features useful for selecting matches by the hypothesis-generation procedure. We do not believe that the problem of grouping image primitives (points, edges, or surfaces) into sets belonging to separate objects will be solved with strictly data-driven (open loop) methods; thus, we do not assume that our segmentation routines are capable of this. We also do not rely on knowing the adjacency of regions in the image or whether regions are occluded, since we feel that this computation is not generally reliable and can be quite complicated to implement. Our recognition algorithm makes few requirements on the capabilities of the feature-extraction algorithm and minimizes its reliance on the results of the feature-extraction algorithm.

There are three distinct components of our VAC system: user-defined modules, the compiler, and the executable recognition program. The user-defined modules consist of models of the objects in our application domain, models of the sensor used to acquire images, image processing and segmentation modules which operate on the input images, and feature modules which define the features that describe the image primitives extracted by the segmentation program. The system is designed so that these modules may be easily added, modified, or removed. The compiler uses the user defined modules to generate the recognition program for the specified models and sensors. It utilizes the sensor model and image processing modules to simulate the imaging and feature extraction process to compile the prior constraints required by the recognition algorithm. This information is combined with the recognition algorithm to form the executable recognition program. Figure 1 shows a schematic of our system.

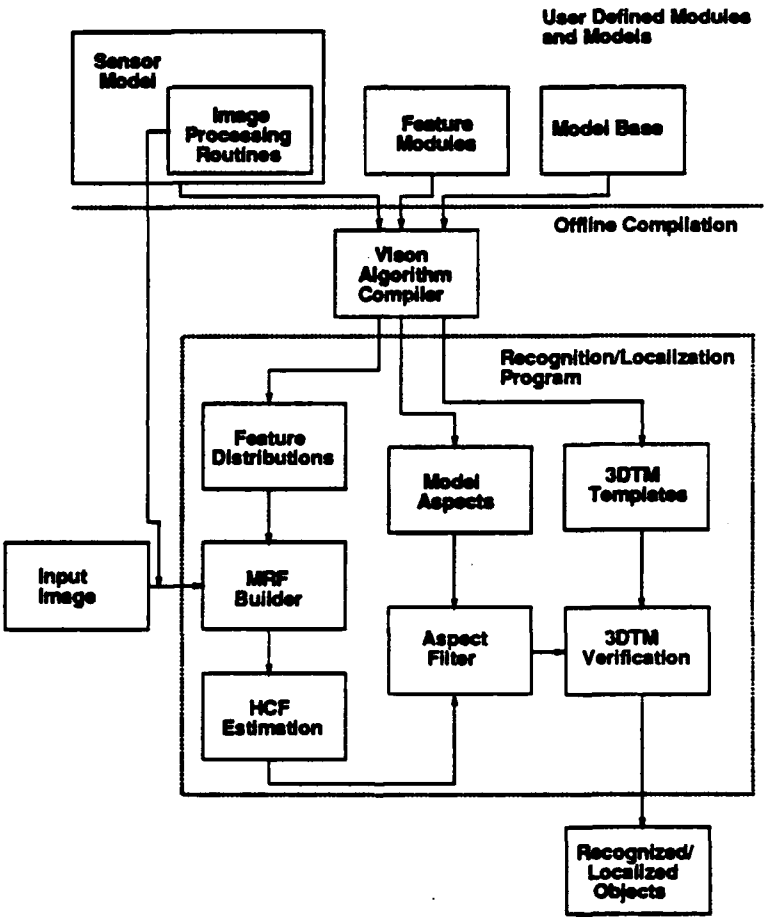


Figure 1: A functional diagram of the Vision Algorithm Compiler system

4. Hypothesis Generation

The first phase of our recognition system is the hypothesize phase. Given a set of primitive features (i.e., planar surfaces or regions) extracted from the input image by a feature-extraction algorithm (i.e., segmentation or edge detection), the hypothesis-generation procedure must produce a set of *possible* model primitive to image primitive matches (hence referred to simply as hypotheses). Optimally, the hypotheses generated include all of the correct correspondences and exclude as many incorrect hypotheses as possible. To exclude incorrect matches, we must apply constraints derived from our prior knowledge. Instead of performing a constrained combinatorial search, we approach hypothesis generation as a process which emits only the most likely hypotheses according to our prior knowledge and constraints.

Using a MRF to represent constraints from our prior knowledge and matches between model features and observed image features, we formulate the search for the most likely hypotheses as a *maximum a posteriori* (MAP) estimate of the MRF. Markov random fields have been typically used for image modeling, edge labeling, and segmentation [CB90, GG87]. In this section, we briefly define MRFs and their relation to the Gibbs distribution. Then, we will detail our formulation of hypothesis generation as a MRF and our compile-time and run-time algorithms to implement the hypothesis generation. Our description of MRFs is based on the description and notation found in [CB90].

4.1. Markov Random Fields

Let X be a set of random variables X_i, each modeled by an element (site) $s_i \in S$. The random variable X_i has a value $\omega_i \in \Lambda$ where Λ is a finite set of labels. Let Ω represent the set of assignments to the random variables X, and $\omega \in \Omega$ represent a particular assignment. We refer to $\omega = (\omega_1, \omega_2, ..., \omega_n)$ as the labeling or assignment to the variables in X.

A neighborhood system N over S is defined as the set of pairs of neighboring sites (s_i, s_j) (the neighbor relation is symmetric and nonreflexive). We can then define N_{s_i} to be the neighborhood of s_i s.t.

$$s_i \notin N_{s_i},$$

$$s_j \in N_{s_i} \Leftrightarrow (s_i, s_j) \in N.$$

Then X is a Markov random field with respect to a probability function P and the neighborhood system N if and only if the following conditions hold:

$$\forall \omega \in \Omega \quad P(X = \omega) > 0,$$

$$P(X_i = \omega_i | X_j = \omega_j, j \neq i) = P(X_i = \omega_i | X_j = \omega_j, s_j \in N_{s_i}). \tag{1}$$

The first condition maintains that each assignment is possible. The second specifies the Markovianity of X with respect to N. The probability distribution of a random variable X_i is only conditionally dependent on the values of the random variables in its neighborhood.

For our application, we are considering many hypotheses simultaneously and wish to choose the most likely subset of these. We can think of the hypotheses as forming a random field of variables with the label set containing the labels **on** or **off**. The hypotheses also display Markovian characteristics. For example, two hypotheses may provide mutual support for each other. If one of them is correct, it is more likely that the other is correct. A dependency also exists between contradicting hypotheses; if one of them is correct, it is unlikely that the other is also correct. These dependencies can be thought of in terms of conditionally dependent probability distributions. With only the definition of MRFs, it is still not clear how to derive or represent the probability distributions or how to perform a MAP estimate of X over a MRF. The next section describes a theorem which will allow us to derive a convenient method for representing the conditional probability distributions which specify a MRF.

4.2. Gibbs-MRF Equality

The Hammersley-Clifford theorem [CB90] equates the joint probability distribution, $P(\omega) = P(X = \omega)$, of MRFs to Gibbs distributions and, thus, provides a simple way to specify a MRF. The theorem states that a random field X is a MRF with respect to a neighborhood system N if and only if there exists a function V such that

$$\forall \omega \in \Omega \quad P(\omega) = \frac{e^{-\frac{1}{T}U(\omega)}}{Z} \tag{2}$$

where T is a temperature constant (controlling the flatness of the distribution) and Z is the normalizing constant for the distribution and

$$U(\omega) = \sum_{c \in C} V_c(\omega) \tag{3}$$

where C is the set of all cliques in N, and $V_c(\omega)$ measures the potential (energy) of clique c under assignment ω. The distribution $P(\omega)$ is called a Gibbs distribution with respect to the neighborhood system N. $U(\omega)$ can be thought of as the energy of the MRF system when $X = \omega$. The minimum energy state corresponds to the most likely state based on the distribution $P(\omega)$.

The conditional probability distributions from Equation 1 can now be computed as

$$P(X_i = \omega_i | X_j = \omega_j, j \neq i) = \frac{e^{-\frac{1}{T}\sum_{c \in C_i} V_c(\omega)}}{\sum_{\omega'} e^{-\frac{1}{T}\sum_{c \in C_i} V_c(\omega')}} \tag{4}$$

where C_i is the set of cliques that contain s_i. ω' is any assignment that agrees with ω everywhere except possibly at s_i.

We are interested in estimating the state of variables X_i of the field X based on some observation or external evidence O_i. Thus, we would like to get the best estimate of ω given observational evidence O. This task can be thought of as finding the MAP estimate of the

posterior distribution $P(\omega|O)$. Using Bayes' rule, and assuming the likelihoods $P(O_i|\omega_i)$ are conditionally independent, we can write the posterior probability distribution as a Gibbs distribution:

$$P(\omega|O) = \frac{P(\omega)P(O|\omega)}{P(O)} = \frac{e^{-\frac{1}{T}U(\omega)}e^{(\sum_{s_i \in S} \log P(O_i|\omega_i))}}{ZP(O)} = \frac{e^{-\frac{1}{T}(U(\omega) - T\sum_{s_i \in S} \log P(O_i|\omega_i))}}{ZP(O)}. \quad (5)$$

This gives us the posterior energy function

$$U(\omega|O) = \sum_{c \in C} V_c(\omega) - T \sum_{s_i \in S} \log P(O_i|\omega_i). \quad (6)$$

The posterior distribution is now in terms of things we may be able to calculate or specify: clique potentials $V_c(\omega)$ and prior distributions for our observations $P(O_i|\omega_i)$. We can find the most likely state of X, based on our observations O, (the MAP estimate of $P(\omega|O)$) by finding the minimum of our (posterior) energy function $U(\omega|O)$ (note that the constant terms Z and $P(O)$ can be ignored). Notice that the $\log P(O_i|\omega_i)$ term in Equation 6 provides the prior bias or penalty term, while the $V_c(\omega)$ term provides the context of the current state ω.

4.3. Formulation of Hypothesis Generation using MRFs

Using the Gibbs-MRF relationship, we can phrase our search for the most likely hypotheses as a MAP estimation problem by defining our prior constraints in terms of clique potentials and likelihoods in the MRF framework. With this formulation, we can apply a MAP estimation procedure to our MRF with the result being the set of hypotheses with the highest probability of occurring based on our prior knowledge and the observed image features.

The following symbols will be used in our formulation of hypothesis generation as a MAP estimate of a MRF:

- R_i is the ith region in the image,
- M_i is the ith surface in the set of surfaces in the model base,
- (R_i, M_j) represents the hypothesized match between the R_i and M_j,
- S is the set of sites of the MRF,
- $s_{i,j} \in S$ is the site that corresponds to (R_i, M_j),
- N^+ is the neighborhood system where $\{s_{i,j}, s_{l,m}\} \in N^+$ indicates that (R_i, M_j) and (R_l, M_m) are consistent hypotheses,
- N^- is the neighborhood system where $\{s_{i,j}, s_{l,m}\} \in N^-$ indicates that (R_i, M_j) and (R_l, M_m) are inconsistent hypotheses,

- $\Lambda = \{ON, OFF\}$ is the set of labels that each site can be assigned,
- $N_{i,j}^{+}$ is the set of consistent neighbors of $s_{i,j}$,
- $N_{i,j}^{-}$ is the set of inconsistent neighbors of $s_{i,j}$, and
- $\vec{f}_{ri} = (f_{ri}^{1}, f_{ri}^{2}, ..., f_{ri}^{n})$ is the feature vector for R_i.

Each site (variable) in the MRF represents a match hypothesis and can be labeled either on or off. Now that the sites and neighborhood system of the MRF are defined, all that is left is to specify the method of generating the neighborhood system, the clique potentials, and the prior distributions.

Each region is described by a set of viewpoint-invariant feature values. For computational reasons, we assume that these features are independent for a given model face. The reason for using multiple features is to provide a concise parameterization (similar to principle components analysis) of the surfaces they describe. Features are only added to provide greater discriminatory capability for the set of model faces. If the features are not independent, then we have redundant features which are not providing new information and should be removed. The independence assumption gives us:

$$\log P(\vec{f}_{ri}|M_j) = \sum_{n} \log P(f_{ri}^{n}|M_j). \tag{7}$$

During hypothesis-generation phase, we need to determine the likelihood that a region in the image arose from the presence of a model face in the image. We model this as the prior distribution

$$P(O_i|s_{i,j} = ON) = P(R_i|M_j) = P(\vec{f}_{ri}|M_j)$$

and can easily calculate it with Equation 7. In other words, the probability that the region resulted from the presence of the model face is the probability that its feature vector would result from the presence of the model face in the scene. To calculate the likelihood that a hypothesis (R_i, M_j) is incorrect, we equate this to the likelihood that R_i actually arose from any of the other model faces:

$$P(O_i|s_{i,j} = OFF) = \sum_{k \neq j} P(R_i|M_k) = \sum_{k \neq j} P(\vec{f}_{ri}|M_k). \tag{8}$$

The priors essentially provide first-order terms of the posterior distribution $P(\omega_i|O_i)$. The higher-order terms are specified in the clique potentials. In this work, we limited ourselves to 1-cliques and 2-cliques. Because of the dynamic nature of our MRF (every image generates a completely different set of sites and neighborhood system), computing the energies of the higher-order cliques would be very expensive. We also feel that 2-cliques are sufficient to model the higher-order constraints as was found by [GLP87] when evaluating the utility of coupled constraints in the tree-search paradigm.

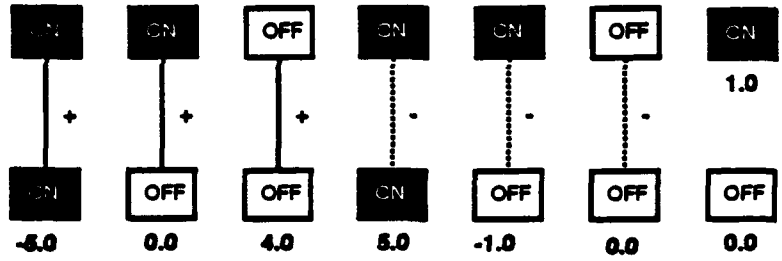


Figure 2: Clique potentials for the six possible configurations of hypothesis labels and neighbor types in 2-cliques and the two possible 1-cliques.

The two neighborhoods over the hypotheses are N^+ for supporting hypotheses and N^- for contradictory hypotheses. The rules that determine the neighborhoods are as follows:

$$\begin{array}{ll} \forall R_i, M_m, M_n \neq M_m & (s_{i,m}, s_{i,n}) \in N^- \\ \forall R_i, R_j \neq R_i, M_m, M_n \neq M_m & (model(M_m) = model(M_n)) \\ & \wedge\ consistent((R_i, M_m), (R_j, M_n)) \ \Rightarrow (s_{i,m}, s_{j,n}) \in N^+ \end{array} \tag{9}$$

where $model(M_m)$ is the model in the model base to which the face M_m belongs, and *consistent*() determines whether the two hypotheses are spatially and geometrically consistent based on the relational features. The above rules essentially state that hypotheses corresponding to the same region are contradictory—we would like one hypothesis per region, and that if two hypotheses are consistent with respect to our prior constraints then they provide mutual support for each other. We are able to compute distributions over second-order features; at this point, however, we have not integrated these distributions into our formulation. Instead, we generate thresholds from these distributions to compute the relation *consistent*(). If it is possible to group regions into those belonging to the same object, the definition of Equation 9 can easily be modified to enforce the required constraints by adding the rule:

$$\begin{array}{ll} \forall R_i, R_j \neq R_i, M_m, M_n \neq M_m & (model(M_m) = model(M_n)) \\ & \wedge\ (object(R_i) = object(R_j)) \\ & \wedge\ \neg consistent((R_i, M_m), (R_j, M_n)) \ \Rightarrow (s_{i,m}, s_{j,n}) \in N^- \end{array}$$

where $object(R_i)$ denotes the identity of the grouped regions containing R_i. In this work, we do not use this constraint since we do not assume that grouping regions belonging to the same object is possible from a purely data-driven approach to segmentation.

The clique potentials (corresponding to $V_c(\omega)$ in Equation 6) used in our experiments appear in Figure 2. For example, the first clique in the figure shows that when a site is on and a consistent (N^+) neighboring site is on, -5.0 is the potential of that 2-clique. The potentials were arrived at experimentally to conform with our sense of consistency and mutual support among hypotheses; of course, a systematic method would be preferred. [She89] has done some work on computing the clique potentials from image models in the image modeling domain. The use of multiple neighborhood systems in one MRF enables us to assign different potentials to cliques based on neighborhood types. This is necessary

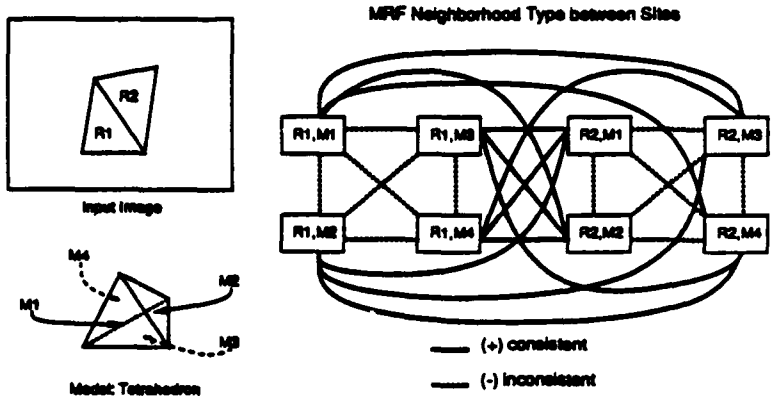


Figure 3: An example MRF produced from a simple scene containing two regions with a model base containing a tetrahedron.

to represent supporting and contradictory relations between hypotheses. This variation from typical MRF applications remains in line with the definition of the Gibbs distribution described in Equation 3.

We can now construct a MRF to perform hypothesis generation. To help the reader visualize a typical resulting MRF, a very simple example is shown in Figure 3. This is an example MRF constructed for an image of a tetrahedron and a model base containing only a tetrahedron. In this case, a site (hypothesized match) is generated for all possible pairs of regions and model faces. The neighborhood relation of these hypotheses is simply that sites for the same region are inconsistent and sites for different regions are consistent.

4.4. Occlusion Sensitivity

When an object surface is partially occluded, the perceived region may no longer satisfy the geometric model constraints; however, at compile time, we can use our sensor simulator to compute the prior distributions of features over images of partially occluded objects. The prior distributions can then account for the constraints from the effects of partial occlusion. Then, if we know a priori that the objects in an image are partially occluded, we can use our partial-occlusion feature distribution during hypothesis generation; however, in general, we are not able to determine whether objects are partially occluded beforehand.

The strategy we adopt is to first assume that everything is unoccluded and attempt to recognize as many of the objects in the image as possible. Then, we attempt to recognize whatever is left with the assumption that the remaining objects are partially occluded. We can parameterize the partial occlusion sensitivity of our system with a single parameter α, which represents the degree of occlusion the recognition program expects in the scene. α has a value between zero and one. For example, a value of one indicates that it assumes every object is partially occluded, while a value of zero indicates that it assumes no objects in the scene are partially occluded. We implement this by using a new feature distribution

$P_\alpha(\vec{f}_r|M_i)$ for each M_i:

$$P_\alpha(\vec{f}_r|M_i) = \alpha P_{occluded}(\vec{f}_r|M_i) + (1-\alpha)P_{unoccluded}(\vec{f}_r|M_i). \quad (10)$$

Our algorithm can iteratively increment its occlusion sensitivity as it attempts to recognize objects until it has accounted for all image regions or until it fails to recognize everything with α equal to one. With the sensitivity high, the recognition process emits more hypotheses for the verification phase since the model constraints become weaker when partial occlusion is assumed. The larger number of hypotheses emitted is mitigated by the ability of the previous iterations to remove some image regions from consideration. Effectively, this approach first recognizes the most easily visible objects in the scene and, then, attempts to recognize the less visible objects.

4.5. Hypothesis-Generation Compile-Time

We have described our formulation of hypothesis generation as a MAP estimation of a MRF. In this section, we describe our system for compiling the prior knowledge from the object models, the sensor model, and the feature modules (see the end of Section 3). In order to generate efficient recognition programs, a favored technique [Goa83] is to precompute as many of the functions as possible and use lookup tables during the execution of the recognition program. Our approach relies heavily on off-line processing since simulated-image generation and segmentation are utilized to generate the prior models.

In addition to efficiency concerns, we also would like a system that is modular with respect to the object-model library, the sensor modality, and features extracted from the input images. When a new object is added to the library, we simply add its module specification as an input to the compiler. The sensor module specifies the appearance simulator for the sensor and the feature-extraction routines which are used to extract primitives from images of the sensor. Given a specified scene and viewing direction, the appearance simulator generates a simulated image from the specified sensor. The feature-extraction routines specify the form of the primitive features available to the recognition program.

The modularity principle also applies to the features which describe the model and image primitives. We have feature modules that have a generic interface to the recognition algorithm. Our recognition algorithm uses first- and second-order features corresponding to unary features over a primitive and relations over pairs of primitives. The compiler takes the specified feature modules and automatically incorporates the use of the features in the resulting recognition program. Currently, we utilize a small set of feature modules. Our first-order features include: region area, maximum second moment, minimum second moment, and maximum axis length. Second-order features include: visibility—are two surfaces visible in an image simultaneously, relative orientation between the surfaces, and maximum distance between surfaces. A higher-order aspect constraint is utilized, after the MRF hypothesis generation, to filter out cliques of consistent hypotheses where multiple (> 2) surfaces are never simultaneously visible in the same image.

The first thing done by the compiler is to build the object model representations to be used by the recognition program. In this work, we use the Vantage solid modeler [BRH+88] which represents models using constructive solid geometry. Our current implementation's sensor modality is range data, and our low-level vision supplies us with segmented planar surfaces. The segmentation algorithm used here is a best-first region growing algorithm which segments the image into planar surfaces. We use an appearance simulator (sensor model) developed by [FNI91] to generate simulated range images. Our feature modules for this system are specified over planar 3-D surfaces.

Once the object models and sensor model are specified, we are ready to compute the prior distributions needed for our MRF hypothesis generation. For each model, we generate a large number of simulated images using our appearance simulator (sensor model). We then process each image using our segmentation routine. This gives us a set of segmented regions from our simulated image. We then calculate the first-order features $\vec{f}_{ri}$ for each R_i in the simulated image and the second-order features over each pair of regions. Since the location of the model in the image is known (this is a simulation), we are able to determine the correspondence of regions to model surfaces. Thus, we can tabulate the feature values for each model face to build the prior distributions $P(f_{ri}^n|M_j)$. The distributions are later smoothed because of the limited number of sample images that we can generate in practice (currently 320 per model). Figure 4 shows an example of an iteration in this process. While generating sample images, we store aspect (based on visible surfaces) information for each view—a surface is visible if the segmentation produces a region corresponding to the model surface. As described in Section 4.4, we also compute prior distributions for the models by generating images where they are partially occluded with at most half of the object visible. For the partial-occlusion distributions, we generate 640 occluded images per model.

Initially, we did not calculate prior distributions using the sensor model but assumed that the feature errors were normally distributed with zero mean and some ad hoc variance. After computing the sampled distributions, it was clear why our MRF hypothesis generation was making mistakes (see Figure 5). The simulated distributions are not normally distributed, and they are biased due to inherent characteristics of the sensor and segmentation algorithm. Additional bias occurs from self occlusion when viewing some object from certain directions. There are secondary modes corresponding to oversegmentations, where a single object surface is segmented into multiple regions. The sensor-modeling approach builds models of the information that the recognition algorithm will actually have available when viewing a known object. This information is dependent on the imaging process and segmentation algorithm, in addition to the known geometric characteristics of the model. An additional benefit of this approach is that model surfaces that are, because of geometric properties of the object, not detectable by the segmentation program will not affect the hypothesis generation. The characteristics of the segmentation of our simulated images correlate strongly with effects from real scene data. Therefore, we are led to believe that this approach to computing priors is useful.

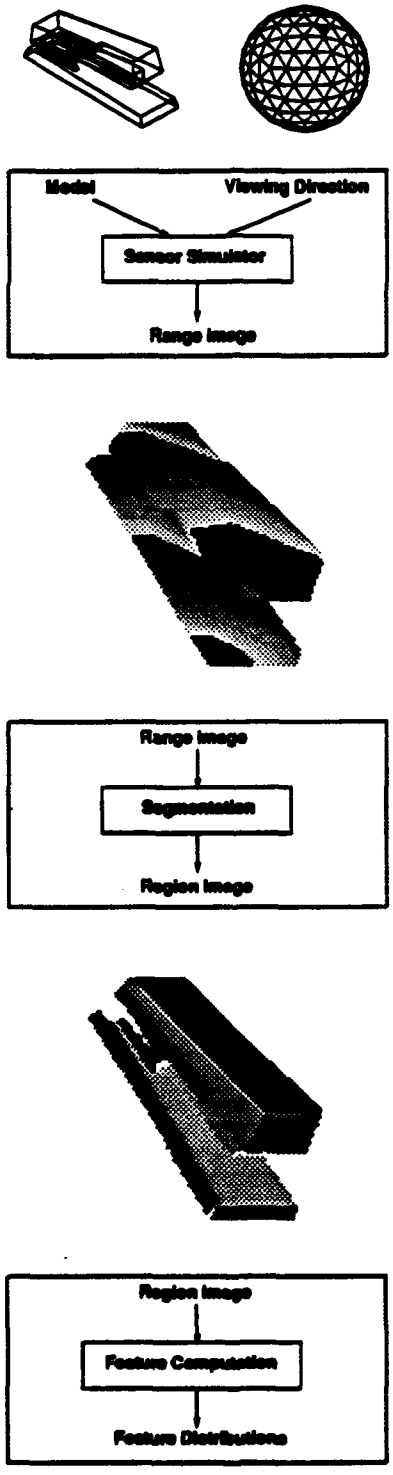


Figure 4: Generation of feature distributions. An object model and viewing direction are selected. The simulator is used to produce a range image of the object which is then segmented into regions which are used to compute the feature distributions.

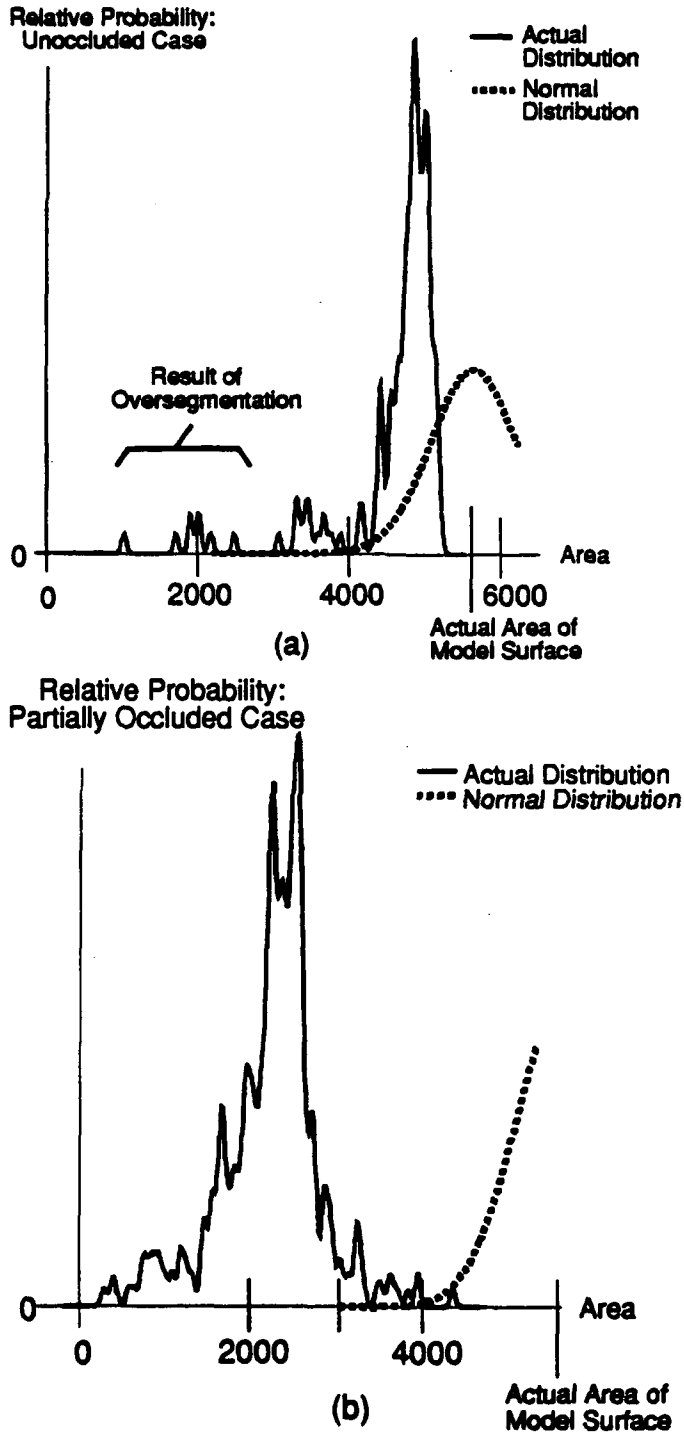


Figure 5: Example distributions of a given feature value (area) over a model face. The distributions were generated by sampling resulting area values from synthetic images of views where (a) the object was unoccluded and (b) the object was partially occluded. A normal distribution centered around the actual model area value is shown to demonstrate the difference between the usual assumption of performance and the actual performance of the segmentation program with respect to the sensor.

4.6. Hypothesis-Generation Run-Time

At run time, the recognition algorithm first segments the image and computes the first-order features over all regions and second-order features over all pairs of regions. The resulting regions and features are converted into a MRF for hypothesis generation as described in Section 4.3, and a sub-optimal estimation procedure (HCF) is used to select the hypotheses. This section describes the MRF estimation procedure and its use in reducing the number of hypotheses.

4.6.1. Use of MRF in hypothesis generation

With the segmented regions, we are able to build our MRF to represent our hypotheses. Using Equations 7 and 8, we first compute the log-likelihoods of the observation of R_i, given that hypothesis (R_i, M_m) is correct, $\log P(O_i|s_{i,m} = ON)$, and incorrect, $\log P(O_i|s_{i,m} = OFF)$. Since scaling the likelihoods, $P(O_i|\omega_i)$ of all labels for a given site by a constant doesn't change the shape of the distribution, we can divide the likelihoods $P(O_i|s_{i,m} = ON)$ and $P(O_i|s_{i,m} = OFF)$ by $P(O_i|s_{i,m} = OFF)$ where they occur in Equation 6. After taking the log of these two terms, we get the log-likelihood ratios:

$$\lambda_{i,m}^{on} = \log \frac{P(O_i|s_{i,m} = ON)}{P(O_i|s_{i,m} = OFF)} = \log P(\vec{f}_{ri}|M_m) - \log(\sum_{n \neq m} P(\vec{f}_{ri}|M_n)) \tag{11}$$

and

$$\lambda_{i,m}^{off} = \log \frac{P(O_i|s_{i,m} = OFF)}{P(O_i|s_{i,m} = OFF)} = 0 \tag{12}$$

which are substituted in place of $\log P(\vec{f}_{ri}|M_m)$ and $\log(\sum_{n \neq m} P(\vec{f}_{ri}|M_n))$ respectively in Equation 6. This simplifies the computations done by the estimation procedure.

When computing $P(\vec{f}_{ri}|M_m)$, we use a lower bound for the probability to cut-off the computation to essentially throw away highly unlikely hypotheses. This lower bound is chosen such that reasonable hypotheses are not thrown away and seems to have very little effect on the final results, while reducing the size of the MRF considerably. With the thresholded hypotheses, we then determine the neighborhood systems N^+ and N^- using the rule specified in Equation 9.

4.6.2. Highest Confidence First Estimation

Once the MRF is created, we wish to find the most likely set of hypotheses based on the constraints of the image. The energy minimization procedure most commonly used with MRFs, simulated annealing, is very slow in practice and cannot be guaranteed to find optimal solutions for large MRFs if one wants to execute it in a reasonable amount of time. Theoretical proofs show that for optimal performance, the annealing schedule will require an

exponential number of steps [GG87] (that's no better than brute-force combinatorial search). To avoid exponential search, we must give up on finding the optimal solution. Fortunately, there is an energy minimization procedure called Highest Confidence First (HCF) [Cho88] which is efficient in practice and, though not optimal, finds good (useful) local minima. HCF was chosen because of its efficiency and evidence of good performance in other applications [Cho88, Coo89].

The general idea of HCF is to do a steepest-descent search in an augmented state space. The label set of the MRF is augmented with the label **uncommitted**. The MRF sites are placed in a heap ordered by the confidence in the site's current label. When labeled **uncommitted**, a site's confidence is defined to be the energy difference between the second-best and best label. Otherwise, the confidence is defined as the energy difference between the current and best label. The site at the top of the heap has its label changed to the best possible (for the current state of the MRF). The confidence values of this site and its neighbor sites are recalculated, and the heap is adjusted. This continues until the site of the top of the heap can no longer decrease the energy of the MRF with a label change. A site may switch labels more than once but may not return to **uncommitted** once it has *committed* to a label. The use of confidence values essentially forces the algorithm to start with sites where a label choice is "most obvious" or has the least competition among labels. The behavior of the HCF search for the most likely set of hypotheses is very similar to the idea of the "focus feature" method of [BHH87]. When there are obvious matches available, the HCF search dives in by turning on the most obvious match first. This creates a ripple effect for matches consistent with obvious matches.

Given a MRF with n sites, the HCF algorithm takes $O(n \log n)$ to initially create the heap and $O(\log n)$ to adjust the heap after modifying a site label, assuming the size of the neighborhoods is constant. In practice, [Cho88] found that the sites are visited slightly more than once on average (consistent with our experience in this application) giving an $O(n \log n)$ performance.

After HCF estimation is completed, the hypotheses of the sites labeled **on** are considered for verification.

5. Localization and Verification

Given a hypothesized set of matches, we must now determine where the object is (**localization**) and whether it is really present in our image or not (**verification**). To succeed, verification depends on an accurate location estimate. If our estimate is poor, we must reject the hypothesis that the object is present at the estimated location. Unfortunately, even slight inaccuracies of location can cause rejection of a hypothesis. In many cases, a slight refinement of the location estimate may be the difference between throwing away a correct hypothesis and recognizing the object. This leads to the question of what defines a good location estimate. The measure used to define the quality of the localization can also be used as the measure for verification. This leads to an algorithm which finds the best location estimate based on the quality measure and then accepts or rejects the hypothesis based on the final value of the quality measure.

Several factors exacerbate the localization problem: we may not have enough constraints from our matches to determine the location of the model accurately, inaccuracies in our region data due to noise and partial occlusion will lead to errors in location estimates, and our objects may vary slightly from the models causing errors in alignment along edges and surfaces. Using primitive matches alone, we are able to get crude estimates of rigid transformation parameters. Thus, localization based on our matches is assumed to be inaccurate but can serve as a good starting point for a local search for the best set of model parameters. What we need is a method which will allow us to drop our model into parameter space at our initial estimate and let the 3-D image data act to pull the model to a more accurate estimate—the essential idea of active contours and snakes [TWK88, KWT87]. Our method for solving this problem, 3-D template matching (3DTM), is described in this section.

5.1. 3-D Template Matching

The general idea of template matching is that we can define a parameterized template to model our object and specify an energy function over the model parameters which relates how closely the model matches the image data. Then, we can perform a search over the parameter space to find the best parameters by minimizing the energy function. Since we are dealing with 3-D images, we define the 3-D template of a model to be a set of points sampled from the surface of the model. Our constraint on the templates is that visible points on the model surface match range data points in the image. The sampled points of the template allow us to efficiently approximate integrals over the surface of the model in the fashion of Finite Element Methods [Zie77, TM91, PS91]. The template of a rigid model is parameterized by rigid-body transformation parameters (rotation and translation). Several researchers have applied similar ideas to model localization (parameter estimation) for 3-D models using range images, most notably [DHI92, TM91, PS91].

The differences between the various methods lie in the techniques and formulation of the estimation. [DHI92, TM91, PS91] use estimation procedures based on dynamics formulations

of the models with respect to image/data forces. As we shall describe a little later, the various methods are similar to *maximum a posteriori* (MAP) estimation of model parameters under some assumed probability distribution of errors.

The definition of the data forces in the dynamics formulations essentially specifies the assumed error distribution of the image data. The relation between the dynamics formulation and MAP estimation techniques is realized by comparing the general method of solving the MAP estimate using a gradient-descent update rule to the update rule derived through the dynamics formulation. Assuming an error distribution of $P(z) \propto e^{-\rho(z)}$ (z is the error) over the points in our model with respect to the image, we can find the MAP estimate by minimizing the energy function

$$E = \sum_{i=1}^{n} \rho(z_i). \tag{13}$$

where z_i is the error of the ith point in the model. By taking the derivative of E with respect to our model (template) parameters q, we get

$$\frac{\partial E}{\partial q} = \sum_{i=1}^{n} \psi(z_i) \frac{\partial z_i}{\partial q} \tag{14}$$

where $\psi(z) = \frac{d\rho(z)}{dz}$. To minimize E, we can use the gradient-descent update rule:

$$\Delta q \propto -\frac{\partial E}{\partial q}.$$

The general form of the dynamics formulation [PS91] is:

$$M\frac{\partial^2 X}{\partial t^2} + C\frac{\partial X}{\partial t} + KX = F$$

where M is the mass matrix, C is the damping matrix, K is the stiffness matrix, F is the force vector, and X is the vector of displacements of the points on the model or the model parameters (i.e., q). The mass matrix can be set to zero to form a first-order system that still has useful dynamics, and K is set to zero for rigid-body parameters. The first-order formulation gives us the dynamic (first-order) update rule for our model parameters q:

$$q_{i+1} = q_i + \Delta t(C^{-1})(F - Kq_i).$$

Thus, in the dynamics formulation, the forces F correspond to Δq in our MAP estimation formulation, and the damping matrix C controls the step size.

Taking $z = \| \vec{x}(q) - \vec{x}_a \|$ (where $\vec{x}(q)$ is the model point and $\vec{x}_a$ is the desired point) and looking at the form of $\psi(z_i)\frac{\partial z_i}{\partial q}$ from Equation 14 with $\frac{\partial z}{\partial q} = \frac{\vec{x}(q) - \vec{x}_a}{z} \cdot \frac{\partial \vec{x}(q)}{\partial q}$, we see that a force with magnitude $f(\vec{x}(q)) \propto z$ corresponds to $\psi(z) = z$. This implies that $\rho(z) = \frac{z^2}{2}$. This is the formula corresponding to a Gaussian distribution, the MAP estimate of which is equivalent to the least-squared error solution. Thus, a dynamics formulation utilizing data

forces with the magnitude of the force proportional to the magnitude of the error vector (as is the case with [TM91, PS91]) would correspond to a least-squared error technique.

In [TM91], a first-order system is used where the forces are proportional to the magnitude of the error. This corresponds to weighted least-mean-square deviation where the formulation of the damping matrix provides a method for computing appropriate scale factors for the step size in the update rule. This is fine for their application since partial occlusion is not dealt with. [DHI92] uses a first-order system which corresponds to minimizing the least-mean-square deviation for the error between model and data points within a specified tolerance while reducing the effect of points outside this tolerance. Their method effectively throws out outliers using a threshold.

For our purposes, we assume that parts of the object surfaces are often occluded. Occlusion can be due to self-occlusion, nearby objects in the scene, and even sensor shadows (visible portions of the scene which don't receive light from the light-striper in a light-stripe range finders). Occluded points are considered to be outliers as are noisy points due to illumination irregularities and sensor error. If outliers are likely, a least-squared-error estimation procedure is not desirable since the estimated parameters will be affected more by noise than the actual data. The shape of the distributions essentially determines how likely outliers are assumed to occur. Least-squares estimates are very sensitive to outliers since all errors are equally weighted proportional to their magnitude. Instead, we would like an estimation procedure which throws out (or gives low weight to) the true outliers. This simply corresponds to a MAP estimate using a distribution where large errors are more likely than in a normal distribution. [DHI92] uses an extreme of such a distribution where large errors are increasingly likely once the error threshold is past.

In this work, we use a *Lorentzian* distribution

$$P(\frac{z}{\sigma}) \propto \frac{1}{1+\frac{1}{2}(\frac{z}{\sigma})^2}$$

to perform the MAP estimate of our model parameters. Using

$$\rho(\frac{z}{\sigma}) = -\log P(\frac{z}{\sigma}) = \log(1+\frac{1}{2}(\frac{z}{\sigma})^2), \tag{15}$$

we can find the MAP estimate by minimizing the energy function of Equation 13. The Lorentzian is similar in shape to the Gaussian distribution, but the tail of the distribution is much larger indicating that outliers are assumed to occur with a higher (relative) probability than in the Gaussian noise model. Figure 6 compares the (unnormalized) Gaussian distribution with the Lorentzian distribution. The important graph is Figure 6(d) which shows the weighting (relative to magnitude) of the error vectors under the Gaussian and Lorentzian distributions. The effect of the Lorentzian is to eventually give zero weight to the true outliers hence improving the estimation of parameters. It is thus, in some sense, providing robust estimate of parameters. Comparing the method of [DHI92] in Figure 7, the shape of the weight function is similar, but the smoothness of the Lorentzian makes the method more stable with respect to the σ parameter (the threshold which changes Delingette's weight

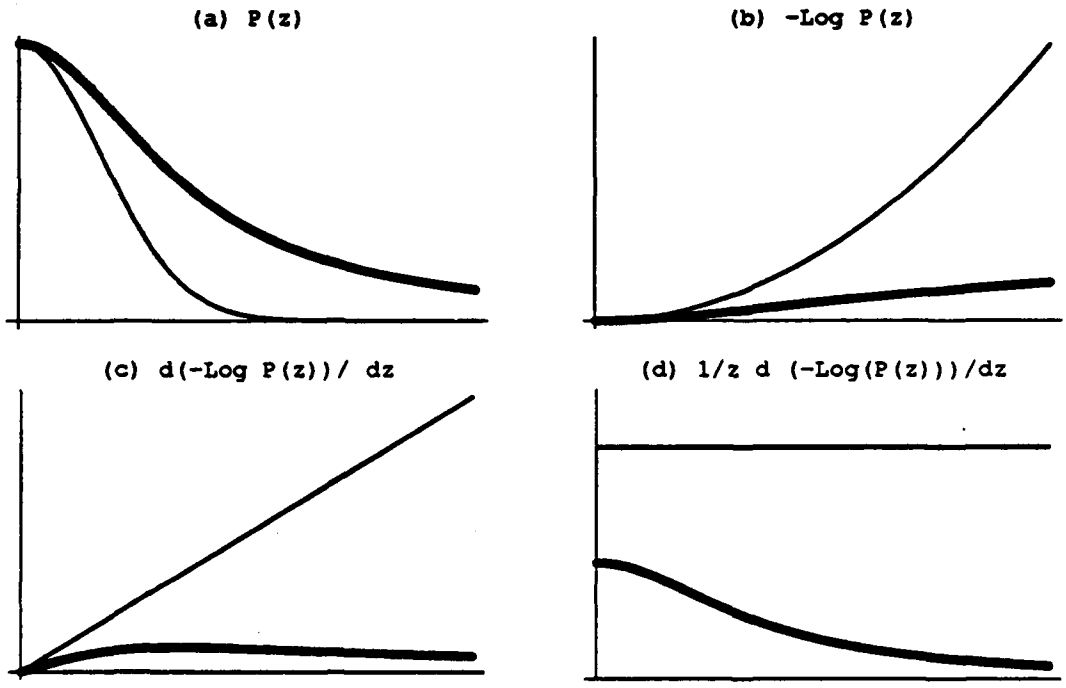


Figure 6: Comparison of Gaussian and Lorentzian distributions and their effect on outliers. The Lorentzian is in bold. (a) Gaussian and Lorentzian distributions, (b) $\rho(z)$ of the Gaussian and Lorentzian distributions, (c) the derivative of $\rho(z)$ which is the magnitude of the "force" corresponding to the data error, (d) the weight of the error vector as a function of the error magnitude for Gaussian and Lorentzian distributions.

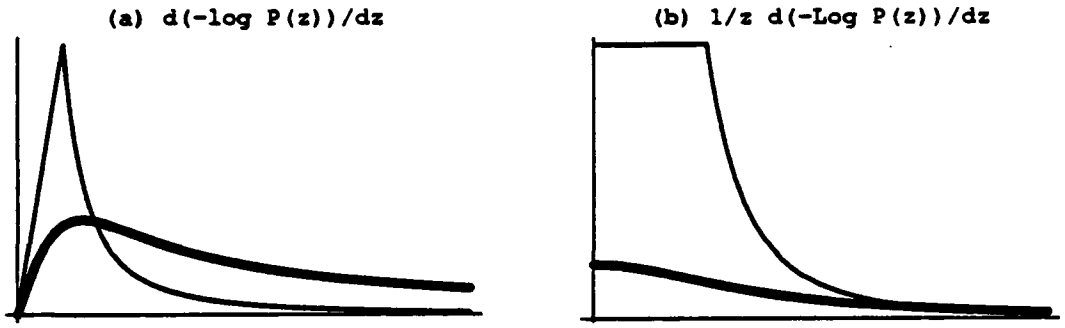


Figure 7: Comparison of the weight function used by Delingette and Lorentzian distributions and their effect on outliers. Lorentzian version is in bold. (a) The derivative of the negative logarithm of the probability distribution (what Delingette terms the weight function for the force), (b) the weight of the error vector for the given magnitude.

function from linear to inverse quadratic); as σ increases, the weight function of [DHI92] forces the solution towards the least-squares solution.

The resulting method is to minimize the function E in our model-parameter space given a starting point for the model parameters. This local search of the parameter space is a MAP estimate of q—the parameters which minimize E. Regardless of the search technique used (i.e., gradient descent, dynamic formulations, conjugate-gradient methods etc.), the time-consuming part of the search will be the function evaluations of E; thus, techniques are needed to make this as efficient as possible.

Having described the general idea of and rationale for our method, we will detail the actual techniques used to implement it. The goal is to improve our model parameter estimate using the range data of our image. We define

$$q = (q_t, q_\theta)^T$$

to be the the vector of model parameters where $q_t = (q_x, q_y, q_z)^T$ is the vector of translation parameters and $q_\theta = (q_s, q_u, q_v, q_w)^T$ is the vector of rotation parameters using the quaternion representation [TM91, FH86]. The function that is minimized is

$$E(q) = \sum_{i \in V(q)} \rho(\frac{z_i(q)}{\sigma}) \tag{16}$$

where $V(q)$ is the set of visible model points for the given model parameters q, $\rho(z)$ is the negative logarithm of the Lorentzian defined in Equation 15, $z_i(q)$ is the error of the ith model point given the model parameters q, and σ is the normalizing factor for the Lorentzian function which specifies the width of the distribution (similar to the standard deviation of a normal distribution).

We define the error to be the distance between the model point and the data point nearest the model point:

$$z_i(q) = \min_{\vec{s} \in D} \| \vec{x}_i(q) - \vec{a} \| \tag{17}$$

where D is the set of three dimensional data points in the image, and $\vec{x}_i(q)$ is the world coordinate of the ith model point transformed using the model parameters q. The calculation of the nearest data point $\vec{a}$ is optimized by using a k-dimensional nearest-neighbor search [FBF77]. This search requires a k-d tree to be built (one time for each image) which takes $O(|D| \log |D|)$ time and takes expected time proportional to $\log |D|$ to find the nearest neighbor of a given point.

The task of localizing 3-D models in complicated scenes using this type of technique introduces a couple of problems. First, in range images, we require the constraint that our localized model not occlude any of the range data that does not belong to the object being located. Secondly, the computation of the visible model points is very expensive since the exact solution essentially requires ray tracing.

The first problem is remedied by using two different values for the normalizing constant σ of Equation 16, depending on the relation between the model point and the nearest point in the range image. When a model point is occluding a data point, the physical constraints of the range image formation are violated. Thus, we expect the probability of this occurrence to be low and use a smaller value for σ to express this. In this work, we assume that a significant portion of the model can be occluded. Thus, when a model point is being occluded by a data point, we use a higher value of σ to express the increased likelihood of this occurrence in our energy function being minimized. Figure 8 shows the shape of the asymmetric probability distribution using the two σ values and an example of how the sign of the error with respect to the viewing direction is used to distinguish the occluding case from the occluded case. In the work reported here, a value of 2 mm for σ was used for occluded model points and 1 mm for occluding model points. This is reasonable since the accuracy of our range data is on the order of 1 mm.

Since only a fraction of the model points are visible from any viewing direction, we only want to include the visible points of the model when computing E. Exclusion of hidden model points is necessary to reduce the number of outliers for our estimate, but it also saves computation costs by reducing the number of points over which we must compute E. For a given pose of the model, we could compute the visible points by performing hidden-point removal or ray tracing. This is inefficient and, in most cases, an overkill. Instead, we use the aspect information computed offline to define an efficient approximation function for determining the visibility of each point. We define aspects to be equivalence classes of visible faces over the set of possible viewing directions. Knowing the current viewing direction, we can determine the aspect of the model and hence the set of visible faces. To determine whether a point is visible, we simply check to see if the face it lies on is visible. This method does not exactly solve the question of whether a point is visible or not since it does not account for possible self-occlusion. However, since we assume there is a significant probability for outliers, we can overcome the effects if a small fraction of points are really not visible from the current viewing direction since they will be considered to be outliers in the estimation. The fact that exact aspects are not necessary in this case lends support for the argument of the utility of aspect representations advocated by Ikeuchi [BFM+92].

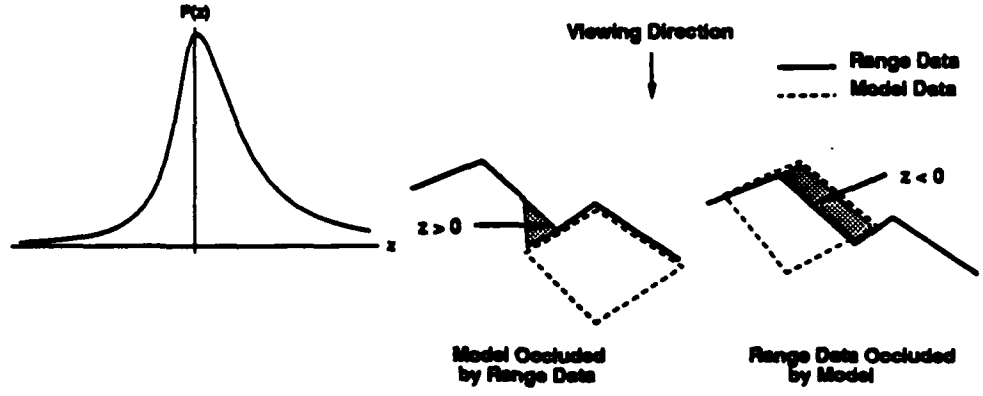


Figure 8: The probability distribution (left) used to enforce the constraint that model points not occlude the range data points ($z < 0$ indicates the model point is occluding the range data) and two cross sections of range data points and model points demonstrating a model occluded by range data and occluding range data.

With the definition of the energy E, we are able to apply any of a number of minimization procedures. Equations for computing the gradient of the model parameters (especially the rotation parameters) can be derived in a manner similar to [TM91]. We utilize a form of gradient-descent search to minimize the energy. We iteratively switch between searching in the gradient direction of the translation parameters and rotation parameters separately. This is necessary because the differences in sensitivity of the parameters create a scale problem. Simply scaling the gradient component of each parameter results in the energy being minimized in terms of the most sensitive parameters only. In addition, we would like to limit the search to a reasonable size hyperellipsoid about our initial guess. To solve these problems, we utilize a bounded line search in the gradient direction and perform this search only over similarly sensitive parameters (i.e., the translation parameters or the rotation parameters) at each iteration. If we know the required accuracy of our pose estimates, we can specify a minimum step size in terms of each parameter and stop the search when a minimum-size step in the gradient direction no longer improves the estimate. In our experiments, the number of gradient-direction steps taken was dependent on the initial starting point but never worse than 25 steps. The minimum step size we used was 1 mm in translation and 1 degree rotation.

There is also a problem with the discontinuities in E produced by using sampled aspects to determine which model points are visible and should be included in the energy computation. If we're searching along a gradient direction and cross an aspect boundary, the search may be prematurely terminated because of a jump in the energy function caused by an increased number of points being visible in the new aspect. In order to reduce this effect, we perform all line searches utilizing the same set of visible points for the entire line search. The

current aspect is computed before each line search and is used to determine point visibility throughout the line search; thus, the energy function is kept smooth throughout the line search. In our experience, this method has reasonable convergence time (in terms of gradient evaluations) with accurate results. Our algorithm is able to converge to good model location estimates from fairly poor (around 4 cm translational error and 20 degrees rotational error for objects of diameter of 5 to 10 cm) starting points.

5.2. Verification Compile Time

The final step for the compilation stage of the VAC is to build the model representation required by the localization/verification step which performs 3DTM.

To build the model representation required by 3DTM, the model is first sampled along each edge and over each surface by projecting a uniformly spaced grid of points onto the surface. A post-processing step removes surface points which lie within the spacing width of an edge point. The resulting sampling is uniform in the sense that no two points are within the spacing width of each other. To compute the visibility of the point, each point is annotated by the list of faces to which it belongs. The list of aspects of the model and the faces associated with each aspect are also included in the model's 3DTM description. Thus, the 3DTM description of the model consists simply of the list of model points and the aspect information.

5.3. Verification Run Time

The hypothesis-generation phase produces a list of cliques of hypothesized primitive matches. The verification phase must determine which of these hypotheses describe objects present in the scene.

The most important step is ordering the cliques so that the correct hypotheses are, on average, close to the top of the list, thus, eliminating the need for verifying incorrect or redundant hypotheses which have regions in common. We order the hypothesis cliques by the average of the likelihood ratios, $P(O_i|s_{i,j} = ON)/P(O_i|s_{i,j} = OFF)$ (see Section 4.3), of their constituent match hypotheses—checking the most likely first. Another consideration is the ability of the clique hypotheses to localize the model. If three surfaces are independently oriented, we can localize the model accurately. If only two surfaces are independent, we have a degree of freedom left for the verification stage to consider. Thus, the more accurately the clique localizes the model, the faster the verification. Thus, the higher cardinality cliques should be verified first. Among cliques of the same cardinality, the cliques are ordered on the basis of their likelihood ratios using the log-likelihood information for each match. The hypothesis cliques are sorted to reflect these considerations.

There is much redundancy in the list of hypothesis cliques. For every 3-clique in the list, there are three 2-cliques and three 1-cliques. These sub-cliques can be eliminated from

consideration once the clique they belong to is accepted by the verification procedure.

The verification procedure takes the ordered list of hypothesis cliques and performs the following set of steps on the first element of the list until the list is empty:

- **Estimate the pose of the hypothesized object using the available matches**
- **Refine the pose estimate by performing 3DTM**
- **Accept or reject the hypothesis based on the value of E found by 3DTM**
- **If the hypothesis is rejected: remove the hypothesis from list**
- **If the hypothesis is accepted: add the model and its estimated parameters to the list of recognized objects, and remove all hypothesis cliques which have regions in common with the current hypothesis**

The first step is complicated by the fact that the hypothesis cliques may have one, two or three matches. The easiest case is when there are three matches that are linearly independent. The pose can be estimated very accurately in this case. An additional test is performed on three cliques to ensure that the three hypotheses are mutually consistent before embarking on the expensive 3DTM localization/verification step.

When there are two matches, there is one degree of freedom in the pose estimate. All that is needed is to match one point on one of the region surfaces to the correct point on the corresponding model surface. We simply use a least-squares solution to an overconstrained match by matching the center of gravity of the two regions with the center of gravity of the corresponding models (essentially four point matches including the normals). Since our 3DTM localization method is robust to poor initial estimates, we can tolerate some error in this estimate. With partial occlusion, significant translation errors can be present with our initial estimate since the error in the center of gravity estimate will be high.

Similarly with one match (or two or three involving linearly dependent surfaces), there are two degrees of freedom that must be resolved to accurately estimate the pose. In this case, we need two point-wise matches between the region surface and the model surface. Again, we use the center of gravity as one point match. The second point match currently used is the maximum inertial vector of the surface offset from the center of gravity. Obviously, this method is not guaranteed to work since partial occlusion can possibly cause a large (greater than 90 degrees) orientation error with this method. A simple solution is to generate a small set of initial estimates with varying orientation and perform 3DTM on each until one is accepted.

Currently, we use an empirical threshold on E to decide whether to accept or reject the hypothesis. This threshold depends on the value of the normalizing factor σ (see Section 5.1). For this work the hypothesis was accepted if the average value of $\rho()$ over all visible points was less than 2.5.

6. Algorithm Complexity

One of the crucial issues in model-based vision is the *scalability* of the algorithm as the size of the model-base increases. In this section, we will analyze the complexity of the algorithms we use to determine whether our system is *scalable*—how does the execution time of our system depend on the size of the model base. This analysis allows us to gain insight into potential bottlenecks and fundamental limitations of our methods.

We use the following variables in our analysis:

- M denotes the number of models in our model base,
- m_{max} denotes the maximum number of primitives (surfaces) for a model in our model base,
- $m = m_{max} \cdot M$ denotes the total number of primitives (surfaces) for the models in our model base,
- n denotes the average number of scene primitives extracted from the input image,
- p denotes the average number of sampled points in a 3DTM model template, and
- d is the number of range data points in the input image.

We sketch the time complexity of our algorithms in terms of the above variables. In practice, the dominant variables are n,m_{max}, d, and p since the model bases we currently deal with are relatively small. For example, d might be 250,000 while M might be 1000 (for an ambitious system). Thus, it is important to derive some bounds based on n,m_{max}, d, and p since this determines if the implementation is feasible for current computer capabilities. However, it is important to determine the effect of the model base size, M, on the execution time since for all practical purposes n,m_{max}, d, and p won't vary significantly if we have a model base of 2 objects or 1000 objects. A good algorithm should be feasible in terms of n,m_{max}, d, and p while scalable in terms of M.

6.1. Compile Time Complexity

The off-line stage is dominated by the image sampling required to compute the prior distributions of the features. For each model, 960 images are generated using our appearance simulator, and each image is processed by the segmentation routine. The execution time of the appearance simulator algorithm is linear in the complexity of the model (the number of surfaces), and its complexity can be denoted as $O(m_{max})$. The segmentation algorithm is only dependent on the image size d, which is constant; thus, we can consider it a constant time operation. To process 960 images of each model, our complexity is $O(960Mm_{max}))$ which reduces to $O(m)$. The constant coefficient for this process is quite large but not critical

since the process is performed off-line. The fact that each model can be compiled separately allows us to add models without recompiling older models and possibly compile the models in parallel.

6.2. Run Time Complexity

The run time stage begins with the segmentation of the image which executes in constant time. Given n regions found by the segmentation algorithm, we need to compute the log-likelihood ratios (Equations 11 and 12) of each hypothesis. This involves computing the first-order feature values of each region ($O(n)$ steps) and looking up the probability from the distribution table for each possible match ($O(nm)$ steps). The result in the worst case is a MRF with nm sites (match hypotheses). To build the neighborhood data structure for the MRF, we must create the neighborhoods N^+ and N^- over the hypotheses using Equation 9. There are $O(m^2)$ connections among hypotheses for each region, giving a total of $O(nm^2)$ connections for the N^- neighborhood. We must also connect the consistent hypotheses belonging to the same model (neighborhood N^+ sites). For each model, there are $O((nm_{max})^2)$ of these connections which require us to test the second-order features for consistency, giving a total of $O(M(m_{max}n)^2)$ connections for the N^+ neighborhood. Thus, we can construct our MRF in

$$O(m_{max}mn^2 + nm^2).$$

Once the MRF is constructed, we must perform HCF estimation on the MRF. We derive an average case bound on the execution of HCF. With nm sites, the HCF algorithm will require $O(nm\log(nm))$ steps to create the heap. To determine the bound on HCF search, we must know the number of sites as well as the number of neighbors of each site. To get a bound on the number of neighbors a site may have, imagine the worst case when each site is connected to all sites sharing the same region, which is at worst the total number of model faces $O(m_{max}M)$, and is connected to all other sites that share the same model $O(m_{max}n)$ (assuming they are all consistent). Thus, each site in our MRF has $O(m_{max}(n+M))$ neighbors. Each iteration of HCF updates the site at the top of the heap and its neighbor sites. Each heap update takes $O(\log(nm))$ steps on average. Thus, each iteration of the algorithm will require $O((n+M)m_{max}\log(nm))$ steps to update the heap. Assuming HCF visits a site a constant (less than 2 in our experience) number of times on average, then the HCF algorithm takes

$$O(nm\cdot(n+M)m_{max}\log(nm)) = O((n^2mm_{max} + nm^2)\log(nm)).$$

After running HCF, we are left with a set of $O(nm)$ active hypotheses in the worst case. We then group these hypotheses into consistent cliques. The consistent cliques exist between hypotheses of different regions but the same model. This gives us $O(M(m_{max}n)^3)$ 3-cliques, $O(M(m_{max}n)^2)$ 2-cliques, and $O(mn)$ 1-cliques giving $O(M(m_{max}n)^3)$ hypothesis cliques to verify. These cliques can also be found in $O(M(m_{max}n)^3)$ time. Combining the costs of

building the MRF, HCF estimation and clique finding, we get a bound of

$$O(M(m_{max}n)^3 + (n^2mm_{max} + nm^2)\log(nm)).$$

In the worst case, the verification phase will check $O(M(m_{max}n)^3)$ hypothesis cliques. Before verification, the cliques are sorted based on their likelihoods; this requires

$$O(M(m_{max}n)^3 \log(M(m_{max}n)^3)).$$

time. For each hypothesis verified, we minimize our energy function (Equation 13) using gradient-descent search. In practice, we can bound the number of iterations needed to find the minimum although it is possible that the solution will be affected. The minimization is then dependent on the cost of computing E. To compute E, we have to compute $\rho(z)$ (Equation 15) for $O(p)$ points on the model. This computation requires us to do a nearest-neighbor search of all d of the range-data points. This computation takes $O(\log d)$ expected time using the k-d nearest-neighbor search algorithm (a k-d tree is built once per image at a cost of $O(d \log d)$). Thus, the verification of a single model is done in $O(p \log d)$ time and the total verification stage is bounded by

$$O(M(m_{max}n)^3 p \log d + M(m_{max}n)^3 \log(M(m_{max}n)^3)).$$

Summing up the bounds for each part of the algorithm we get an algorithm bounded by

$$O(M(m_{max}n)^3 p \log d + M(m_{max}n)^3 \log(M(m_{max}n)^3) + (n^2mm_{max} + nm^2)\log(nm)).$$

Fortunately, the only instance where we might possibly reach this bound is when an object in the scene does not belong to our model base. In Section 7, we provide an example of the actual performance of our algorithm on a scene containing *no* known objects.

Assuming that the model complexity is bounded (m_{max} and p constant) and knowing that the number of image points d is constant and $n < d$, we find the complexity in terms of the size of the model base to be

$$O((M + M^2)\log M) = O(M^2 \log M).$$

Our analysis of course is a very high upper bound of the actual performance because of the following two factors:

- each region will never be assigned m hypotheses unless the model base is extremely regular (i.e., all object surfaces geometrically similar), and
- the estimate for the number of N^+ connections in the MRF, $O((nm_{max})^2)$, is very high since the model consistency relations can only be satisfied between regions that are spatially close.

The M^2 term is unsatisfactory. Note that this term results from the use of N^- neighborhoods. It might be possible to define the clique potentials in such a way that this neighborhood constraint is not necessary. Removing it would still only give us an $O(M \log M)$ complexity. It looks unlikely that it is possible to get a linear complexity using the hypothesize-and-verify paradigm. The reason is that (without a miracle) the number of hypotheses generated will be proportional to the number of models in the model base (the number of hypotheses is $O(M)$). Unless we blindly verify every hypothesis, we will need to do some ordering of the hypotheses to get effective pruning of the hypothesis through verification. This ordering step will give us the $O(M \log M)$ complexity regardless of our hypothesis selection method.

Feasibility depends on the variables n, m_{max}, d, and p. Considering only these variables, we get a complexity of

$$O((m_{max}n)^3(p \log d + \log(m_{max}n)))$$

which is comparable to other algorithms. The cubic term results from the use of surfaces, which require 3 correspondences to determine the pose, as our only primitive features. The terms $\log d$ and $\log(m_{max}n)$ are rather insignificant while p is the most significant.

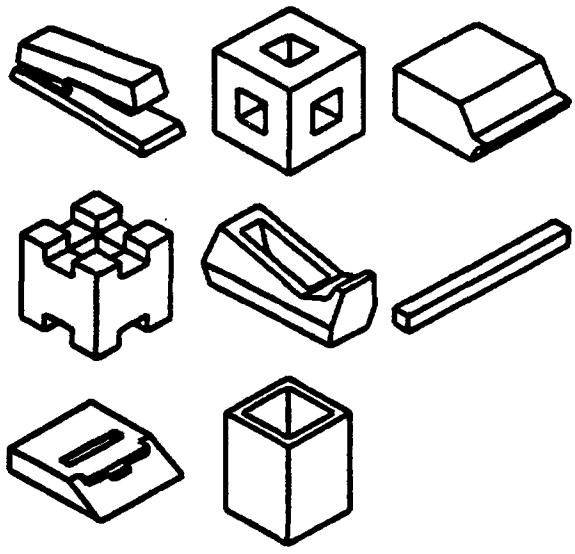


Figure 9: The model base of 8 objects used during this experiment (from left to right, top to bottom): stapler, hole-cube, rolodex, castle, tape dispenser, stick, note dispenser, and pencil box.

7. Recognition Results and Experiments

To evaluate the performance of our system, we are interested in the effectiveness of the hypothesis generation and the accuracy of our localization. Our strategy relies on the fact that successful recognition of objects in the image can be used to reduce the number of hypotheses that require verification. Thus, our localization algorithm must be robust (insensitive) to inaccurate initial estimates and occlusion.

We begin by looking at the convergence results of 3DTM in a few example images. In Figure 10, we have three examples of the convergence of the localization estimates in sample range images. The first images show a complicated example of localizing the tape dispenser. Note that there is significant occlusion and extraneous data points caused by the roll of tape in the dispenser as the tape is not part of the object model. Even with the large rotational (around 20 degrees) and translational error (around 3 cm) of the initial estimate, the model is accurately localized (the estimated error of location is 2 mm translation and 2 degrees rotation). The second pair of images shows the localization of a cylinder as it is picked up by a human hand. The algorithm accurately localizes the grasped object despite the occlusion caused by the hand. The third pair shows the localization of the rolodex after it has been picked up by a hand. There are additional outliers caused by the lack of range data points on the right side of the rolodex which is under sensor shadow. Again, the occlusion caused by the hand and sensor shadow has little effect on the result.

To get an idea of the usefulness of the Lorentzian distribution for accurate localization, we present three examples comparing the resulting location estimates of our method using

Gaussian distributions and Lorentzian distributions. These examples appear in Figure 11. These examples demonstrate the convergence ability of 3DTM given estimates that correspond to significant rotation and translation errors and aspect changes. The first example shows the performance in localizing the hole-cube which is partially occluded. The result of the Gaussian estimate is close but has a noticeable rotation error resulting from the the slight occlusion of the bottom corner of the cube. The Lorentzian estimate, however, is very accurate and is not noticeably biased by the occlusion. The second example is more convincing. This example shows a pencil box which is also only slightly occluded but contains a significant amount of self-occlusion (see the inside face of the box). Because of our simple method of using aspect face visibility to determine the visibility of points in the estimation, self-occluded faces which are partially visible activate points which are actually not visible. These points have the same effect as the points occluded by other objects. In this example, the least-squares estimate finds a rotation and translation estimate which balances the error of the visible and invisible points. The Lorentzian estimate is able to lock on to the actual visible points to accurately locate the pencil box. The third example shows the performance in localizing the rolodex which is partially occluded by the cylinder resting on top of it. In this case, the least squares estimate is greatly affected by the presence of the cylinder on top causing a rotational error of about 8 degrees compared to about 2 degrees for the Lorentzian estimate. As discussed in Section 5.1 and as you can see in Figure 11, the least-squares solutions are noticeably occlusion sensitive while the results using the Lorentzian distribution are relatively insensitive to occlusion. In experiments, we have found that the initial (rough) location estimates can be perturbed by a few cm in translation and around 20 degrees in rotation without affecting the resulting solution.

To evaluate the performance of our hypothesis-generation system, we are really interested in the number of hypotheses verified since the verification stage is the expensive component of our algorithm. Our strategy relies on the fact that successful recognition of objects in the image can be used to reduce the number of hypotheses that require verification. We present statistics on the number of hypotheses emitted by the hypothesis-generation phase using our technique on several sample images.

These experiments were conducted using the model-base of 8 objects shown in Figure 9. All of the objects are approximated by polyhedral models. Three of the objects in this group (hole-cube, castle, and stick) are geometric objects used in the Assembly Plan from Observation research of [SI91, IS91]. These three objects are interesting because of their high degree of symmetry and similarity of first-order and relational features (for example, the large faces of the hole-cube and castle have very similar area and the adjacent faces on these objects are all at right angles).

In each of the examples, three iterations of the hypothesize-and-verify algorithm were performed with a value of 0.0, 0.5 and 1.0 respectively for the occlusion-sensitivity parameter (see Section 4.4). We performed three iterations to get an idea of the verification cost when using the unoccluded distribution, the partial-occlusion distribution and a mixture of the two distributions. At the end of each iteration, regions that correspond to recognized objects are removed from consideration for the next iteration. Frequently, hypotheses that were not

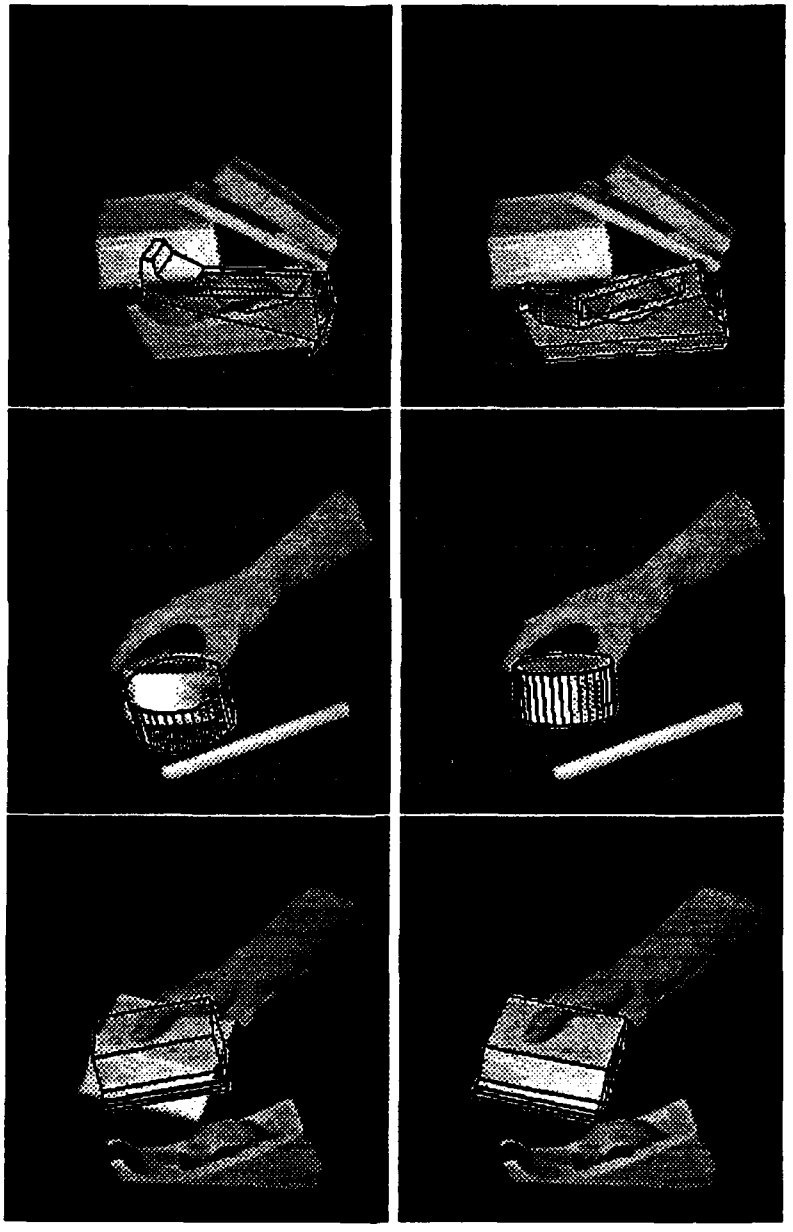


Figure 10: Example of convergence results for the localization problem using 3dtm: initial model locations (left images), and final localized model locations (right images).

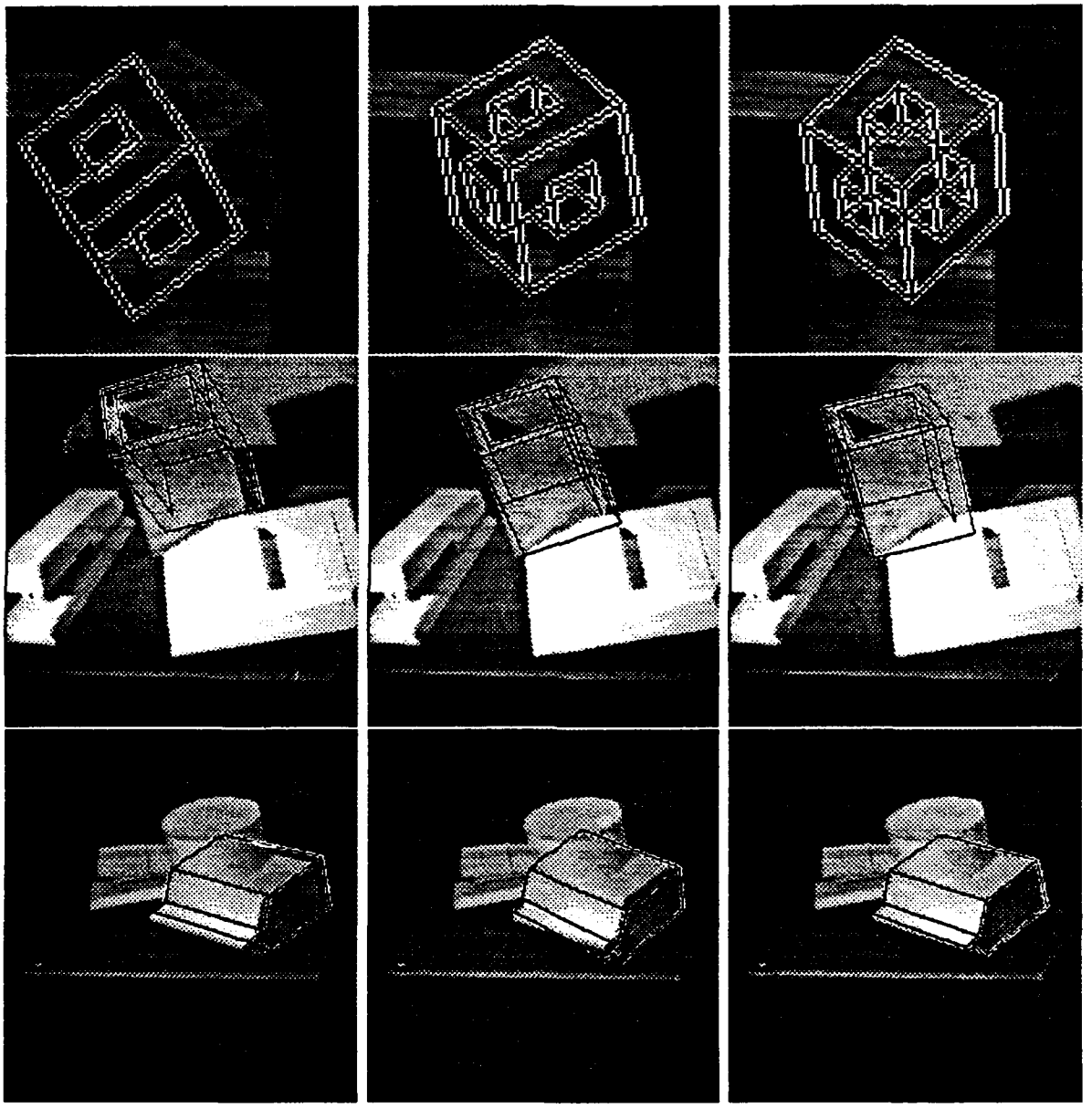
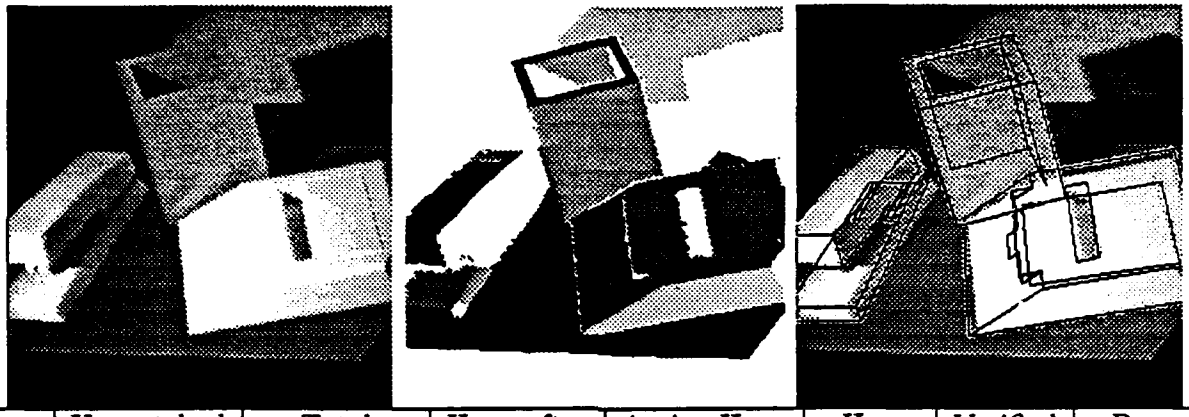


Figure 11: Comparison of the results of the least-squares formulation and the robust formulation of the error distribution on three localization problems: initial model location (left), final localized model location using least-squares (middle), and final localized model location using robust formulation (right).



Occlusion Parameter	Unmatched Regions	Total Hypotheses	Hyps after Threshold	Active Hyps after HCF	Hyp Cliques	Verified Cliques	Recognized Models
0.0	16	2672	98	24	48	14	pencil box note-dispenser stapler
0.5	3	501	11	0	0	0	none
1.0	3	501	8	0	0	0	none
					total verifications	14	

Figure 12: Example recognition/localization results: intensity image (left), segmented regions (middle), wire frame overlay of models (right). The table lists the number of hypotheses generated and verified for each iteration of the algorithm using the specified occlusion parameter for computing the log-likelihoods.

accepted by the verification procedure are generated in the next iteration. These rejected hypotheses are not reverified.

For scenes containing known models, we are interested in how well the ordering of hypotheses reduces the number of verifications required. On the tested examples, the first few hypotheses describe objects that were actually in the scene. When this occurs, the number of hypotheses verified is greatly reduced. What's left after the known objects are recognized are hypotheses for regions that do not correspond to known objects. Unfortunately, we must verify all of the hypotheses generated for these regions since the verification will not succeed.

The first example demonstrates the performance in spite of slight occlusion. The right side of the stapler is under a sensor shadow, the stapler and the note dispenser are both partially out of the image, and the pencil box is slightly occluded by the note dispenser. In addition, the top surface of the stapler was oversegmented into two regions. The algorithm was able to recognize the pencil box, note dispenser, and stapler with an occlusion parameter of 0.0. For this case, where there is only slight occlusion, the algorithm performs well, with only 14 verifications required. The results of the localization show that the CAD models of the objects have noticeable inaccuracies, but the location estimates are still quite useful.

The second recognition example, Figure 13, shows the same objects in a different configu-

Occlusion Parameter	Unmatched Regions	Total Hypotheses	Hyps after Threshold	Active Hyps after HCF	Hyp Cliques	Verified Cliques	Recognized Models
0.0	13	2171	98	20	32	20	note-dispenser pencil-box
0.5	7	1169	49	5	7	7	none
1.0	7	1169	44	7	15	15	none
					total verifications	42	

Figure 13: Example recognition/localization results: intensity image (left), segmented regions (middle), wire frame overlay of models (right). The table lists the number of hypotheses generated and verified for each iteration of the algorithm using the specified occlusion parameter for computing the log-likelihoods.

ration with significant partial occlusion. The pencil box is partially occluded by some pencils and by a sensor shadow. The note dispenser is partially occluded by a sensor shadow which is caused by the pencil box, and the stapler is almost completely hidden by the pencil box. The recognition program was able to locate the pencil box and the note dispenser despite the amount of partial occlusion. Recognition of the note dispenser is quite surprising though. The fact that it was recognized in the first iteration was because the unoccluded distributions actually include oversegmentations which is equivalent to partial occlusion from the point of view of the recognition program. Unfortunately, the stapler was not recognized.

Figure 14 shows the results of an experiment which recognized objects in a sequence of 3 images. Objects were added to the scene in between images. In this test, the objects recognized in the previous image were used as initial hypotheses in the current image and were relocalized and verified using 3DTM. In between images, the previously recognized objects were slightly moved by the action of adding the new objects to the scene. 3DTM was able to relocalize these objects and, thus, reduce the number of regions for consideration. The statistics show how the complexity of the recognition process can be greatly reduced when dealing with images that change slowly over time. This example is from the assembly plan from observation research of [SI91], where image sequences are used for automatic robot assembly planning.

Finally, we give an example of an image containing no known models. This gives us an idea of what the actual worst case performance might be in terms of the number of hypotheses verified. In this case, each verification fails, which means that all hypothesis cliques generated by the hypothesis-generation phase must be verified using 3DTM.

The number of hypotheses verified is exacerbated by the regions due to background or unknown objects. In fact, the majority of the incorrect hypotheses were due to unknown regions satisfying binary relations of model faces, and their number was increased greatly by the symmetry of the models. The symmetry of the model base is the leading culprit with regards to the generation of redundant (unnecessary) hypotheses for verification.

Our prototype recognition program was implemented in Common Lisp for a Sun 4 workstation. The approximate execution time was 2 minutes for the segmentation, 2-5 minutes for building the MRF, and 2-5 seconds to perform HCF and order the hypotheses for verification. The prototype of 3DTM takes approximately 1-5 minutes to perform localization (much of which is image input and output). We have not concentrated on making the implementation efficient. Instead, we have opted for fast a development environment in which to test our ideas.

These results demonstrate the ability of our system to generate useful hypotheses and localize the hypothesized models in the scene using only region primitives and simple unary features and binary relations on the regions. They also demonstrate an effective technique for recognizing and localizing occluded objects in non-trivial scenes.

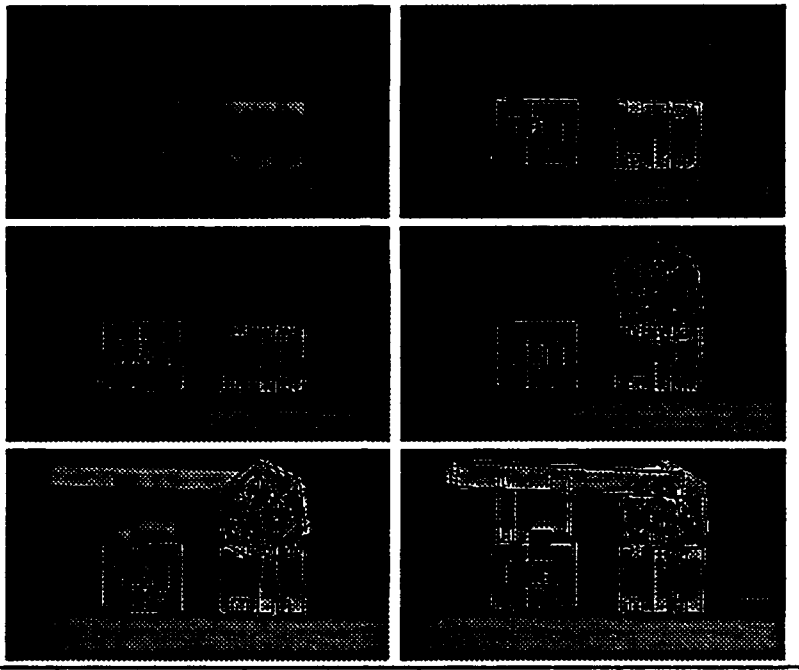
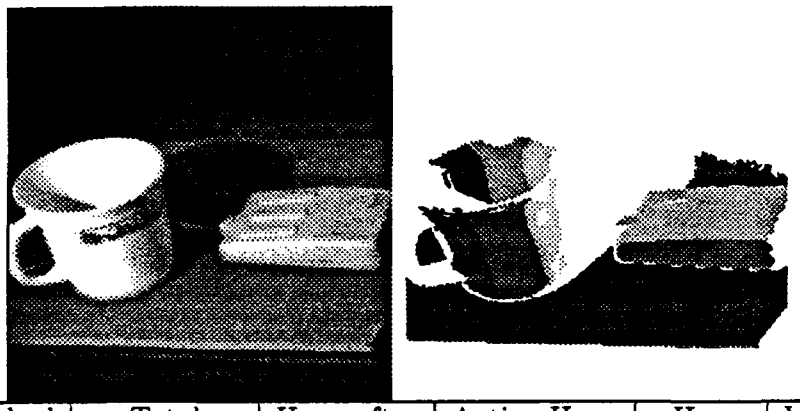


Image	Occlusion Parameter	Unmatched Regions	Total Hypotheses	Hyps after Threshold	Active Hyps after HCF	Hyp Cliques	Verified Cliques	Recognized Models
1	0.0	8	1336	69	19	35	7	hole-cube castle
	0.5	4	668	56	12	12	12	none
	1.0	4	668	51	11	11	11	none
2	0.0	10	1670	42	15	77	3	hole-cube
					relocation of hole-cube and castle		2	
	0.5	3	501	21	5	5	5	none
	1.0	3	501	21	0	0	0	none
3	0.0	15	2505	92	13	15	13	castle
					relocation of 2 hole-cubes and castle		3	
	0.5	7	1169	187	21	30	10	stick
	1.0	5	835	93	4	4	4	none
						total verifications	70	

Figure 14: Example of recognition/localization results on a sequence of 3 images: intensity image with previously recognized models overlayed (left column), wire frame overlay of recognized models (right column). The table lists, for iteration on each image, the number of hypotheses active at each phase of the recognition algorithm. The table lists, for iteration on each image, the number of hypotheses active at each phase of the recognition algorithm.



Occlusion Parameter	Unmatched Regions	Total Hypotheses	Hyps after Threshold	Active Hyps after HCF	Hyp Cliques	Verified Cliques	Recognized Models
0.0	14	2338	105	24	45	45	none
0.5	14	2338	132	19	42	42	none
1.0	14	2338	144	9	20	20	none
					total verifications	107	

Figure 15: Example of hypothesis-generation results for an image containing no known objects: intensity image (left), segmented regions (right). The table lists the number of hypotheses generated and verified for each iteration of the algorithm using the specified occlusion parameter for computing the log-likelihoods. In this case since no known objects exist in the scene, all hypotheses had to be verified.

8. Conclusions

We have presented an approach to 3-D object recognition and localization which utilizes realistic models of the sensor and segmentation algorithm and explicitly accounts for the effects of partial occlusion in the hypothesis-generation and localization/verification phases.

We have introduced the sensor-modeling approach for hypothesis generation for object recognition. The sensor-modeling approach has the following advantages:

- it provides realistic (accurate) constraints for "optimal" hypothesis generation by explicitly modeling the effects of the sensor, the segmentation algorithm, the geometry of the objects (including self-occlusion), and feature detectability,
- it can be used to model the effect of partial occlusion through simulation,
- it builds prior models are robust with respect to segmentation capabilities, and
- real world constraints about likely viewing directions for particular objects can be utilized by the sensor-modeling approach to improve the hypothesis-generation performance.

The MRF formalism combined with sensor modeling provides a framework for "optimal" hypothesis generation with respect to the prior knowledge from our sensor model. HCF estimation provides an efficient and effective method of performing the estimation over our MRF. In experiments, our sensor-modeling approach to hypothesis generation has demonstrated the ability to reduce the recognition time by accurately selecting hypotheses and "optimally" ordering the hypotheses for verification.

We have presented an approach to 3-D object localization in range images which is robust to occlusion of the object being located. The examples of our algorithm's performance demonstrate its convergence ability and accuracy compared to results obtained using least-squares estimation when presented with objects that are partially occluded or near extraneous objects in the scene. The algorithm is applicable for fine localization of objects in an object-recognition system as well as object tracking in image sequences. It may also have applications in image registration and automatic model acquisition.

An important contribution of this paper is the demonstration of the utility of robust estimation methods for the localization problem. Another advantage of our localization method is that it only requires a crude estimate of the model parameters and then utilizes the actual image data to fine-tune the estimate without the necessity of any explicit matching process (i.e., connecting model points to specific data points [Low89]) which usually relies on information obtained from higher-level processes. The technique used by [Low89] is based on minimizing the least-squared error of visible model edges and scene edges. This works when there are no occlusions but may be inaccurate with significant occlusion. Our formulation of the error using the Lorentzian probability distribution enables us to compute

accurate localizations even when significant portions of the object are occluded. The use of the Lorentzian distribution also reduces the dependence on thresholds for outlier detection.

The ability to drop the model into the image and let go without the necessity for high-level matching is an essential difference between our method and [Low90]. It is closer in spirit to the idea of active contour models and snakes inspired by [KWT87, TWK88]. In this work (as well as other object-recognition research), we can consider a hypothesis-generating recognition algorithm to be the equivalent of Witkin's human operator which initializes the "snake" or "active contour". It is also interesting to consider (as Lowe noted in [Low90]) the merging of recognition and motion tracking capabilities through the localization algorithm. This produces an algorithm for recognition of image sequences which first performs verification of the hypotheses that the objects from the previous image are in the same location and then executing the entire recognition process on any unexplained image data.

9. Future Work

We now would like to point out some areas for further research and ideas for improving our system.

First, the domain of applicability of the system must be extended to handle a wider variety of object representations, sensor modalities, and types of primitive features. The current implementation deals only with polygonal objects—a quite limited set of objects considering real world applications. However, we believe that the addition of higher-order parameterized surfaces is more of a problem for the segmentation procedure than a problem for the recognition algorithm. This is because the higher-order surfaces provide more distinct features which provide stronger constraints for the recognition module. The addition of higher-order surfaces would not impact the 3DTM localization and verification method since it makes no assumptions about the model except that its surface is sampled. The integration of objects with more complicated parameterizations such as studied by [Low89] is a problem that also needs to be addressed. Adding parameterized deformations would also be helpful for localizing objects which have varying dimensions or for cases where the object model is inaccurate.

There is also much work to be done concerning our derivation of MRF priors and clique potentials. We don't have a nice way to calculate the clique potentials and have to rely on experimentation to determine appropriate settings. Our prior distributions are generated by taking sample images with only two degrees of freedom, the orientation of the object. This is a first-order approximation to computing the priors. Obviously, scale will also affect the performance of the segmentation algorithms. We still need to determine how much other degrees of freedom affect the distributions. Methods for utilizing the prior probabilities of the binary features to specify the neighbor relationship between pairs of hypotheses would be interesting to investigate. A more realistic approach to simulating partial occlusion needs to be investigated to generate more effective prior distributions for hypothesis generation.

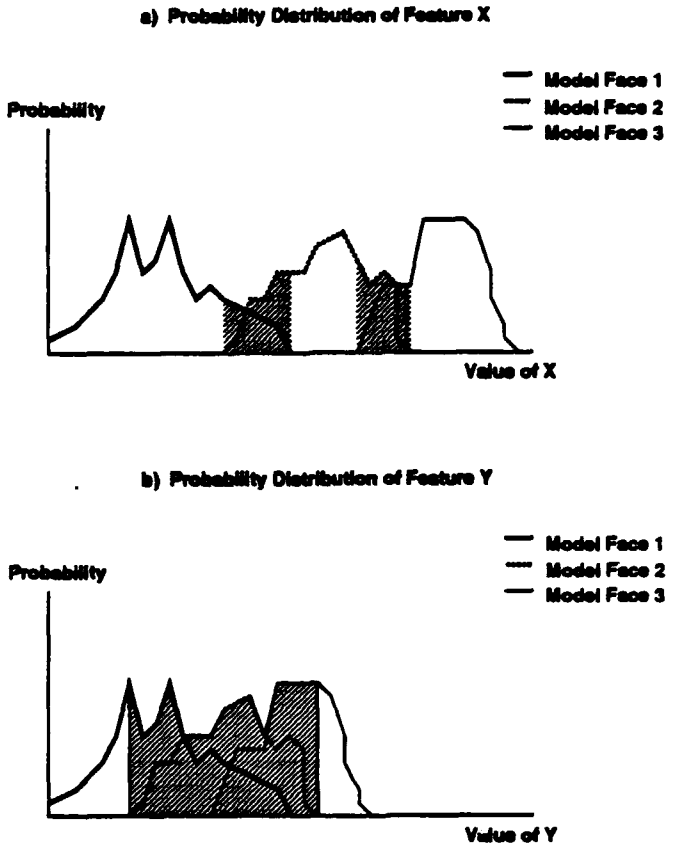


Figure 16: Feature Saliency examples: (a) a salient feature which has little overlap among the model faces in the model-base, (b) a non-salient feature which has little discriminatory capability in the particular model-base

One constraint which is not currently required for our system is the spatial-occupancy constraint of solids. This essentially considers the possibility that two recognized objects intersect in a way that is not visible from the viewing direction. We can determine whether any pair of recognized objects intersect using the model locations. This was done by [SI91] and can easily be applied here. The models used in this work were not capable of creating this type of ambiguity.

Another optimization concerns the idea of feature saliency [Fly92]. Feature saliency refers to the ability of a given feature to resolve ambiguities or separate the possible matches. Feature saliency is dependent on the model base. For example, if every object in our model base had a different color, we would want to use the color feature as our first test when computing the likelihoods. By first testing the color, all candidate hypotheses belonging to the incorrect model would be eliminated immediately. This is the fail first strategy used in search techniques—to find out as early as possible if you are wrong before spending too much time on a particular solution. Our feature distributions provide a nice way of computing saliency. The saliency may be defined as a measure of the amount of overlap of the feature value's distribution among the set of model faces. A very salient feature has little

overlap between the distributions for the set of model faces (see Figure 16). If we order our calculation of feature values from most salient to least, we can optimize the computation of hypothesis likelihoods which involves a large number of operations. When computing the likelihood of each hypothesis using Equation 7 and summing the log-likelihoods in order of decreasing saliency, we are more likely to be able to filter the hypotheses without having to compute the likelihood over every feature since the salient features will automatically eliminate many of the extremely unlikely hypotheses.

Model symmetry is a major source of redundant verification of hypotheses. In our experiments, symmetries accounted for a large percentage of the incorrect hypotheses that had to be verified. Symmetry has a multiplicative effect on the number of hypotheses generated for regions that are mutually consistent with model constraints. We should be able to take advantage of symmetries to reduce the execution time of our algorithm by reducing the number of model surfaces that are considered for matches and the number of hypotheses that must be verified.

Manual model acquisition is very tedious and often inaccurate. Our sensor-modeling approach can be thought of as a form of automatic model acquisition where the imaging process is simulated using CAD models and an appearance simulator. Replacing the appearance simulator with a real sensor would speed up the compilation stage, but the sampling of random views with respect to the object would have to be mechanized. Integrating automatic model acquisition with the sensor-modeling approach appears to be quite promising and should be investigated.

To improve the 3DTM localization algorithm, the first thing that comes to mind is its susceptibility to local minima. In our system, local minima may occur in cluttered scenes when the initial estimate is very far from the actual object. For cases where local minima is a problem, higher-level image features such as edge information become useful in building a system that has large basins of attraction. The edge information can be used to define "long range" forces as done in the work of [DHI92]. Further work must be done to determine the limits of the amount of partial occlusion that our method can tolerate.

There are several optimizations available to reduce the execution time of 3DTM. We can partition the computation of E over parallel processors since each point of the model is independent of another. To speed up the nearest-neighbors search, we can remove the range-data points which are sufficiently far from the initial model estimate. In our current implementation, much of the execution time of the recognition program is taken up by redundant overhead computations in 3DTM. These computations can easily be amortized over all of the verifications done per image. Minimization algorithms other than our gradient-descent method must be more thoroughly investigated.

Our verification procedure still relies on experimental thresholds for accept/reject decisions. The work done by [GH91] provides a more rigorous formulation for the verification decision thresholds and may be applicable here.

With respect to segmentation capabilities, undersegmentation, resulting from similarly

shaped and oriented surfaces adjoining in the scene, is still problematic. The solution likely requires higher-level feedback from the recognition program to split unrecognized surfaces. In some cases, our iterative recognition scheme could be modified to solve this problem by removing data corresponding to recognized objects and resegmenting the remaining image data.

Acknowledgments

The authors would like to thank Paul Cooper for a helpful discussion concerning the use of MRFs for object recognition, Martial Hebert for helpful comments and the use of his segmentation code, and Sing Bing Kang, Kevin Lynch, Heung-Yeung Shum and Fredric Solomon for providing helpful comments on this paper.

References

[BCK90] Ruud M. Bolle, Andrea Califano, and Rick Kjeldsen. A scalable and extendable approach to visual recognition. Technical Report RC 15477, IBM Research Division, T.J. Watson Research Center, 1990.

[BFM+92] Kevin Bowyer, Olivier Faugeras, Joe Mundy, Narendra Ahuja, Charles Dyer, Alex Pentland, Ramesh Jain, and Katsushi Ikeuchi. Workshop panel report: Why aspect graphs are not (yet) practical for computer vision. *Computer Vision, Graphics and Image Processing: Image Understanding*, 55(2):212–218, 1992.

[BHH87] Robert C. Bolles, Patrice Horaud, and Marsha Jo Hannah. 3DPO: A three-dimensional part orientation system. In Martin A. Fischler and Oscar Firschein, editors, *Readings in Computer Vision: Issues, Problems, Principles, and Paradigms*, pages 355–359. Morgan Kaufmann, 1987.

[BM92] Paul J. Besl and Neil D. McKay. A method for registration of 3-d shapes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14(2):239–256, 1992.

[BRH+88] P. Balakumar, Jean-Christophe Robert, Regis Hoffman, Katsushi Ikeuchi, and Takeo Kanade. VANTAGE: A frame-based geometric modeling system. Technical report, Carnegie Mellon University, 1988.

[CB90] Paul B. Chou and Christopher M. Brown. The theory and practice of Bayesian image labeling. *International Journal of Computer Vision*, 4:185–210, 1990.

[Cho88] Paul Bao-Luo Chou. *The Theory and Practice of Bayesian Image Labeling.* PhD thesis, Department of Computer Science, University of Rochester, 1988. Technical Report 258.

[Coo89] Paul Cooper. *Parallel Object Recognition from Structure (The Tinkertoy Project)*. PhD thesis, Department of Computer Science, University of Rochester, 1989. Technical Report 301.

[DHI92] Herve Delingette, Martial Hebert, and Katsushi Ikeuchi. Shape representation and image segmentation using deformable surfaces. *Image and Vision Computing*, 10(3):132–144, 1992.

[Fan90] Ting-Jun Fan. *Describing and Recognizing 3-D Objects Using Surface Properties*. Springer-Verlag, New York, 1990.

[FBF77] J.H. Friedman, J.L. Bentley, and R.A. Finkel. An algorithm for finding best matches in logarithmic expected time. *ACM Transactions on Mathematical Software*, 3(3):209–226, 1977.

[FH86] O.D. Faugeras and Martial Hebert. The representation, recognition, and locating of 3-d objects. *IJRR*, 5(3):27–52, 1986.

[FJ91] Patrick Flynn and A.K. Jain. Automatic generation of recognition strategies using cad models. In *IEEE Workshop on Directions in Automated CAD-Based Vision*. IEEE, 1991.

[Fly92] Patrick J. Flynn. Saliencies and symmetries: Toward 3d object recognition from large model databases. In *Proceedings of Computer Vision and Pattern Recognition*, pages 322–327. IEEE, 1992.

[FNI91] Yoshimasa Fujiwara, Shree Nayar, and Katsushi Ikeuchi. Appearance simulator for computer vision research. Technical Report CMU-RI-TR-91-16, Carnegie Mellon University, 1991.

[GG87] Stuart Geman and Donald Geman. Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images. In Martin A. Fischler and Oscar Firschein, editors, *Readings in Computer Vision: Issues, Problems, Principles, and Paradigms*, pages 564–584. Morgan Kaufmann, 1987.

[GH91] W. Eric L. Grimson and Daniel P. Huttenlocher. On the verification of hypothesized matches in model-based recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13(12):1201–1213, 1991.

[GLP87] W. Eric L. Grimson and Tomas Lozano-Perez. Localizing overlapping parts by searching the interpretation tree. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 9(4):469–482, 1987.

[Goa83] Chris Goad. Special purpose automatic programming for 3-d model-based vision. In *Proceedings ARPA Image Understanding Workshop*, 1983.

[HCK89] S.A. Hutchinson, R.L. Cromwell, and A.C. Kak. Applying uncertainty reasoning to model based object recognition. In *Proceedings of Computer Vision and Pattern Recognition*, pages 541-548, 1989.

[Hut88] Daniel P. Huttenlocher. *Three-Dimensional Recognition of Solid Objects from a Two-Dimensional Image.* PhD thesis, Massachusetts Institute of Technology, 1988.

[IH91] Katsushi Ikeuchi and Ki Sang Hong. Determining linear shape change: Toward automatic generation of object recognition programs. *Computer Vision, Graphics and Image Processing: Image Understanding*, 53(2):154–170, 1991.

[IK88] Katsushi Ikeuchi and Takeo Kanade. Automatic generation of object recognition programs. *Proceedings of IEEE Special Issue on Computer Vision*, 76:1016–1035, 1988.

[IS91] Katsushi Ikeuchi and Takashi Suehiro. Towards an assembly plan from observation : fine localization based on face contact constraints. Technical Report CMU-CS-91-168, Carnegie Mellon University, 1991.

[KK91] Whoi-Yul Kim and Avinash C. Kak. 3-d object recognition using bipartite matching embedded in discrete relaxation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13(3):224–251, 1991.

[KWT87] Michael Kass, Andrew Witkin, and Demetri Terzopoulos. Snakes: Active contour models. *International Journal of Computer Vision*, 2(1):322–331, 1987.

[Low85] David G. Lowe. *Perceptual Organization and Visual Recognition.* Kluwer Academic Publishers, 1985.

[Low89] David Lowe. Fitting parameterized 3-d models to images. Technical Report 89-26, Computer Science Department, University of British Columbia, 1989.

[Low90] David G. Lowe. Integrated treatment of matching and measurement errors for robust model-based motion tracking. In *Proceedings of Computer Vision and Pattern Recognition*, pages 436–440. IEEE, 1990.

[LW88] Yehezkel Lamdan and Haim J. Wolfson. Geometric hashing: A general and efficient model-based recognition scheme. In *Proceedings of International Conference on Computer Vision*, pages 238–249. IEEE, 1988.

[PM] B. Parvin and G. Medioni. A constraint satisfaction network for matching 3d objects. In *International Conference on Neural Networks*, pages 281–286, ??

[PS91] Alex Pentland and Stan Sclaroff. Closed-form solutions for physically based shape modeling and recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13(7):715–729, 1991.

[She89] David Sher. The ergodic Mondrian model: a source of markov random fields. Technical Report 89-03, State University of New York at Buffalo, 1989.

[SI91] Takashi Suehiro and Katsushi Ikeuchi. Towards an assembly plan from observation : task recognition with polyhedral objects. Technical Report CMU-CS-91-167, Carnegie Mellon University, 1991.

[SIK92] Kosuke Sato, Katsushi Ikeuchi, and Takeo Kanade. Model based recognition of specular objects using sensor models. *Computer Vision, Graphics and Image Processing: Image Understanding*, 55(2):155–169, 1992.

[SM92] Fridtjof Stein and Gerard Medioni. Structural indexing: Efficient 3-d object recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14(2):125–145, 1992.

[TM91] Demetri Terzopoulos and Dimitri Metaxas. Dynamic 3d models with local and global deformations: Deformable superquadrics. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13(7):703–714, 1991.

[TWK88] Demetri Terzopoulos, Andrew Witkin, and Michael Kass. Constraints on deformable models: Recovering 3d shape and nonrigid motion. *Artificial Intelligence*, 36:91–123, 1988.

[YCH89] Alan L. Yuille, David S. Cohen, and Peter W. Hallinan. Feature extraction from faces using deformable templates. In *Proceedings of Computer Vision and Pattern Recognition*, pages 104–109, 1989.

[Zie77] O. C. Zienkiewicz. *The Finite Element Method.* McGraw-Hill, 1977.